# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**JNANASANGAMA, BELAGAVI-590018**



# PROJECT REPORT

**“Real Time Chat Application”**

*Submitted by*

**PRAJAKTA MISHRA(1CR21IS112)**

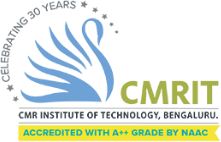
**RACHNAKUMARI (1CR21IS123)**

**Under the guidance of,**

**Prof. Partha Chattopadhyay**

**Assistant Professor,**

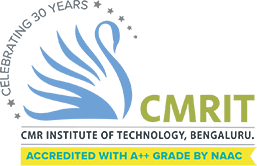
**Dept. of ISE**



#### DEPARTMENT OF INFORMATION SCIENCE& ENGINEERING CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

### 2023-2024



**DEPT. OF INFORMATION SCIENCE & ENGINEERING**

Certificate

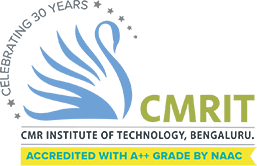
This is to certify that the AngularJS and NodeJS Project work entitled “**Real Time Chat Application**” has been carried out by **Prajakta Mishra (1CR21IS112) and Rachna Kumari (1CR21IS123)** bonafide students of CMR Institute Of Technology, Bengaluru in partial fulfillment for the award of the Degree of **Bachelor of Engineering in Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year **2023-24**. It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the report deposited in the departmental library. This project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

|  |  |
| --- | --- |
| **-----------------------------** | **------------------------** |
| **Signature of Guide** | **Sign. of HOD** |
| **Prof. Partha Chattopadhyay**  **Assistant Professor** | **Dr. Jagadishwari V**  **Professor & HOD** |
| **Department of ISE** | **Department of ISE** |
| **CMRIT, Bengaluru** | **CMRIT, Bengaluru** |

**Viva**

Name of the examiners **Signature with date**

1. .



# ACKNOWLEDGEMENT

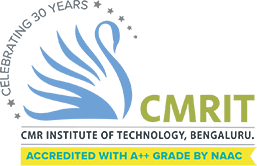
Any work of significance requires a great deal of effort and time put into it. But a factor of even greater importance is efficient guidance and encouragement. In spite of all my dedicated work, this project would not have been possible without continuous help and guidance provided by people who gave their unending support right from when this idea was conceived.

I would like to thank to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant co- operation and support throughout this project.

I would like to thank **Dr. Jagadishwari V,** Professor & HOD **,** Department of Information Science and Engineering, CMRIT for her constant guidance and support during this period.

I would like to thank my guide, **Prof. Partha Chattopadhyay, Assistant Professor,** Department of Information Science and Engineering, CMRIT for her constant guidance that helped me in completing the project work successfully.

Last but definitely not the least I would like to thank **My Family** and **Friends** who have always supported me in every path of the project work.



# DECLARATION

We, the students of V semester from Department of Information Science and Engineering, CMR Institute of Technology, Bangalore declare that the project work entitled "**Real Time Chat Application**” has been successfully completed under the guidance of Prof. Partha Chattopadhyay, Assistant Professor, Dept. of Information Science and Engineering, CMR Institute of technology, Bengaluru. This project work is submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in Information Science and Engineering during the academic year 2023-2024. The matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bangalore

Date:07/03/2024

Team Members:

|  |  |
| --- | --- |
| **Prajakta Mishra, 1CR21IS112** |  |
| **Rachna Kumari, 1CR21IS123** |  |

### TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **S No** | **Content** | **Page No** |
| 1 | Abstract | 6 |
| 2 | Technology used in project | 7 |
| 3 | Implementation | 8 |
| 4 | Screenshots | 9-10 |
| 5 | Future Enhancements | 11-12 |
| 6 | Conclusion | 13 |
| 7 | References | 14 |

## ABSTRACT

The Chat Application project is a web-based communication platform developed using Node.js, Socket.IO, and React, aimed at facilitating real-time messaging between users. The project prioritizes simplicity, scalability, and user experience, offering features such as instant messaging, user authentication, and chat rooms. Leveraging Node.js for server-side operations ensures efficient handling of client-server communication, while Socket.IO enables bidirectional, real-time data exchange between clients and the server, ensuring seamless connectivity across devices. The application's frontend is built using React, providing a responsive and intuitive user interface that enhances user engagement. Challenges encountered during development, such as managing user sessions and optimizing performance, were addressed through iterative development and collaborative problem-solving. The Chat Application project exemplifies proficiency in modern web development technologies, emphasizing real-time communication and user interaction. Future iterations may focus on additional features, security enhancements, and scalability improvements to meet evolving user needs and industry standards. As technology continues to evolve, the project remains adaptable and responsive to emerging trends and user feedback, ensuring continued relevance and effectiveness in the digital communication landscape.

**TECHNOLOGY USED IN PROJECT**

* Node.js is a server-side JavaScript runtime environment that enables developers to run JavaScript code outside the browser. It is built on Chrome's V8 JavaScript engine and is well-suited for building scalable and efficient server-side applications. In the context of the chat application, Node.js is utilized to handle server-side logic, such as managing client connections, processing chat messages, and broadcasting messages to connected clients. Its event-driven, non-blocking I/O model makes it ideal for real-time applications like chat, allowing for concurrent connections without sacrificing performance or scalability.
* Socket.IO is a JavaScript library that facilitates real-time, bidirectional communication between web clients and servers. It abstracts away the complexities of WebSocket-based communication and provides a simple yet powerful API for building real-time applications. In the chat application, Socket.IO is crucial for handling events such as message transmission, connection establishment, and disconnection. Its features like automatic reconnection, multiplexing, and broadcasting streamline the development of real-time applications and ensure reliable communication between clients and the server.
* HTML, CSS, and JavaScript are the core technologies of the World Wide Web, responsible for creating the structure, styling, and interactivity of web pages. HTML provides the semantic structure of the chat application's user interface, defining elements such as input fields and message display areas. CSS is used to style the HTML elements, controlling their appearance, layout, and branding. JavaScript adds dynamic behavior and functionality to the user interface, handling events like message sending and receiving. Together, these technologies form the client-side interface of the chat application, providing a responsive and interactive user experience.
* Client-server communication is the process of exchanging data between a client (e.g., web browser) and a server over a network. In the chat application, client-server communication occurs over WebSocket connections established using Socket.IO. Clients send messages to the server, which broadcasts them to all connected clients in real-time. This bidirectional communication enables instant messaging between users, enhancing user engagement and interaction.

#### IMPLEMENTATION

1. **User Interface Design**:
   * The chat application focuses on creating a visually appealing and user-friendly interface using HTML, CSS, and JavaScript. React components could be utilized to enhance the UI further, creating modular and reusable UI elements. Styling could be achieved using CSS or CSS frameworks like Tailwind CSS to ensure a modern and responsive layout.
2. **Real-time Messaging**:
   * Real-time messaging functionality is the core feature of the chat application, facilitated by Socket.IO. WebSocket connections are established between the server and clients to enable instantaneous communication. Messages are transmitted in real-time, with the chat UI updating dynamically to display incoming and outgoing messages.
3. **Integration with External APIs (Optional)**:
   * While the current implementation does not integrate with external APIs, you could consider integrating with services like Giphy API to enhance the chat experience. This would allow users to send GIFs in chat messages, adding an interactive element to conversations.
4. **User Interactions**:
   * User interactions within the chat interface are managed using event listeners and DOM manipulation in JavaScript. Users can send and receive messages, view chat history, and interact with other users in real-time. Event-driven programming is employed to handle user interactions and update the UI accordingly.
5. **Responsive Design**:
   * The chat application is designed to be responsive, ensuring optimal display across various devices and screen sizes. Media queries and responsive design techniques could be employed to adapt the layout and styling of UI components for desktops, tablets, and mobile devices.

#### SCREENSHOTS

#### 

#### 

#### Login page

#### 

#### 

#### 

#### 

#### FUTURE ENHANCEMENTS

#### 

#### Enhanced Recommendation System:

#### Implement a recommendation system within the chat app to suggest relevant chat rooms or topics based on users' interests, past interactions, and preferences.

#### Utilize machine learning algorithms or collaborative filtering techniques to provide personalized recommendations for users, enhancing their chat experience.

#### Content Expansion:

#### Expand the range of chat topics and categories available within the app to cater to a broader audience and accommodate diverse interests.

#### Introduce curated chat rooms or discussion groups focused on specific themes or subjects to encourage engagement and participation among users.

#### Advanced Search and Filtering Options:

#### Enhance the search functionality within the chat app to allow users to easily find and discover relevant messages, conversations, or users.

#### Implement advanced search features such as keyword-based search, filtering options by date or user, and sorting options to streamline the chat navigation experience.

#### Multi-device Sync:

#### Enable synchronization of chat messages and user preferences across multiple devices, allowing users to seamlessly transition between devices without losing their chat history or preferences.

#### Implement features such as real-time message synchronization, read/unread message status synchronization, and cross-device notifications to ensure a consistent chat experience.

#### Accessibility Improvements:

#### Improve the accessibility of the chat app by implementing features such as keyboard navigation, screen reader support, and high-contrast themes.

#### Conduct accessibility audits and usability testing to identify and address potential barriers to access for users with disabilities or impairments, ensuring inclusivity and usability for all users.

#### Security Enhancements:

#### Strengthen the security of the chat app by implementing measures such as end-to-end encryption for message transmission, user authentication, and authorization mechanisms to protect user data and privacy.

#### Introduce security features to detect and prevent common security threats such as spam, phishing attacks, or malicious content within the chat environment.

## CONCLUSION

In summary, the chat application has made significant strides in providing users with a seamless and interactive communication platform. Leveraging Node.js, Socket.IO, and HTML/CSS/JavaScript technologies, the project has demonstrated proficiency in real-time messaging, user interface design, and client-server communication.

Through the implementation of real-time messaging using Socket.IO, users can engage in instant communication, sending and receiving messages in real-time. The user interface, crafted with HTML, CSS, and JavaScript, prioritizes usability and responsiveness, ensuring a smooth and intuitive experience across devices.

While the current iteration of the chat application excels in its core functionalities, there are several opportunities for future enhancement and expansion. Potential avenues for improvement include implementing advanced features such as message encryption for enhanced security, integration with external APIs for additional functionality, and enhancing accessibility features to cater to a broader audience.

Looking ahead, the chat application has the potential to evolve into a comprehensive communication platform, offering users a rich and immersive chatting experience. With continued development and innovation, the chat application is poised to become a valuable tool for fostering connections and facilitating meaningful conversations among users.

#### REFERENCES

#### Socket io documentation: <https://socket.io/docs/>

#### Node.js Documentation: <https://nodejs.org/en/docs/>

#### HTML Documentation: <https://developer.mozilla.org/en-US/docs/Web/HTML>

#### CSS Documentation: <https://developer.mozilla.org/en-US/docs/Web/CSS>

#### JavaScript Documentation: https://developer.mozilla.org/enUS/docs/Web/JavaScript

#### 