

A PROJECT REPORT

ON

**CHAT APPLICATION**

Submitted By:

Abhishek Kumar-‘Q’-2115000021

Advik Saxena - ‘Q’ – 2115000084

Harshit Sharma - ‘W’ – 2115700003

Garvit Dewan – ‘W’ -- 2115700002

***In partial fulfillment for the award of the degree of***

# Bachelor of Engineering

**IN**

Computer Science

# BONAFIDE CERTIFICATE

Certified that this project report **“Chat Application”** is the bonafide work of “**Abhishek Kumar, Advik Saxena, Harshit Sharma, Garvit Dewan”** who carried out the project work under my/our supervision.

**SIGNATURE**

|  |  |
| --- | --- |
| **SIGNATURE**  **HEAD OF THE DEPARTMENT**  **Computer Science and Engineering** | **SUPERVISOR**  **Technical Trainer**  **T & D Department** |

# ACKNOWLEDGEMENT

The project work in this report is an outcome of continuous work over a period and draws intellectual support from various sources. We would like to articulate our profound gratitude to all those people who extended their wholehearted cooperation and have helped us in completing this project successfully.

We are thankful to our mentor Mr. Ankit Arora for teaching and assisting us with the new technology and guiding us at every step, it wouldn’t have been possible for us to finish the project in such a short period if it were not for his motivation. I would also like to thank all the faculty members who directly and indirectly contributed in the completion of our project.

# ABSTRACT

Mini project is the requirement for all engineering students to complete their Bachelor of Engineering degree at the GLA University, Mathura. This project is very important for us since it complements both the academic and professional aspects of engineering education. It exposed us to practical experience and an actual working environment, where we were able to develop our skills and capabilities, as well as enhance our intellectual and emotional personalities. The Mini Project also provides strong linkages between university industries that shall pave opportunities for "smart partnerships" and industrially driven research. The outcomes of the EIT are mainly based on the assessment covering the company's and university's evaluation will provide feedback for students’ performance after 75% completion of their engineering study. The remarks from the companies on the students will be very helpful for the university to have continuous quality improvement, especially on curriculum practiced.

**Table Of Content**

**CHAPTER 1: Introduction**

* 1. Problem Statement
  2. Objective

CHAPTER 2: Technology Used

* 1. React Js
  2. Express Js
  3. Mongo DB
  4. Chakra UI

CHAPTER 3: System Design and Implementation

* 1. Chat app or client-side
  2. Chat Server Engine
  3. Web socket

CHAPTER 4 RESULTS

4 screenshots

4.1 Setting up the Connection

4.2 Chatting Demo

CHAPTER 5 Conclusion

CHAPTER 6 References

**INTRODUCTION**

In today's fast-paced digital era, communication plays a pivotal role in connecting individuals and facilitating collaboration. As communication methods evolve, the need for efficient and secure messaging platforms becomes increasingly apparent. One such solution that addresses this need is a Chat Application for Record File.

* 1. Problem Statement
* This project is to create a chat application with a server and users to enable the users to chat with each other’s.
* To develop an instant messaging solution to enable users to communicate with each other
* The project should be very easy to enable even a novice person to use it
  1. Objective
* **Simplicity and Ease of Use:** Create a user-friendly messaging platform with a straightforward interface for effortless communication.
* **Basic Messaging Functionality:** Develop a messaging app that allows users to exchange text messages and basic multimedia content, such as images, to facilitate communication.
* **Efficient Contact Management:** Implement a system for users to manage contacts easily, enabling quick and hassle-free messaging with friends and acquaintances.
* **Expressive Communication:** Include a collection of emojis to enable users to express themselves more vividly during conversations.
* **Real-Time Messaging:** Enable instant message delivery and receipt, providing a smooth and responsive chatting experience.

**Technology Used**

* 1. **REACT JS**

React.js, commonly known as React, is a JavaScript library for building user interfaces, particularly for single-page applications where user interactions and updates occur without requiring a full page reload. Developed and maintained by Facebook, React has gained widespread popularity for its declarative approach to building interactive and reusable UI components.

* 1. **Express JS**

Express.js, commonly referred to as Express, is a minimal and flexible web application framework for Node.js. It is designed to simplify the process of building robust and scalable web applications and APIs by providing a set of essential features and utilities. Developed by TJ Holowaychuk and currently maintained by the Node.js foundation, Express.js has become one of the most popular choices for building server-side applications in the JavaScript ecosystem.

* 1. **Mongo Db**

MongoDB is a popular open-source NoSQL database management system that falls under the category of document-oriented databases. It is designed to provide a flexible, scalable, and high-performance solution for handling large volumes of unstructured or semi-structured data. MongoDB stores data in a format known as BSON (Binary JSON), which is a binary-encoded serialization of JSON-like documents.

* 1. **Chakra UI**

Chakra UI is a popular React component library that provides a set of flexible and accessible UI components for building modern web applications. It is designed to make it easier for developers to create visually appealing and responsive user interfaces while adhering to best practices for accessibility and customization.

**System Design and Implementation**

**Designing and implementing a chat application involves several components, including frontend, backend, databases, and communication protocols. Here's a high-level overview of the system design and implementation steps for a chat application:**

1. Requirements Gathering and Analysis

Understand the specific requirements of the chat application:

Types of users (regular users, admins, moderators, etc.).

Features required (one-on-one chat, group chat, multimedia sharing, etc.).

Expected traffic and scalability needs.

Security and privacy considerations.

2. System Architecture Design

a. Frontend:

Choose a technology stack (JavaScript frameworks like React, Vue.js, Angular) for the client-side application.

Design the user interface (UI) for the chat application, including features like message threads, user profiles, and notifications.

b. Backend:

Select a backend technology stack (Node.js, Python with Django or Flask, Ruby on Rails, etc.).

Implement RESTful APIs or use WebSocket for real-time communication between clients and the server.

Implement user authentication and authorization mechanisms (JWT, OAuth, etc.).

Develop features for message handling, storage, and retrieval.

Consider using a message broker (like RabbitMQ, or Kafka) for handling message queues in a distributed system.

c. Database:

Choose a database system (SQL or NoSQL) based on scalability, performance, and data requirements (MySQL, PostgreSQL, MongoDB, etc.).

Design database schemas for users, messages, conversations, and other relevant entities.

Optimize the database for efficient storage and retrieval of messages.

d. Communication Protocols:

Implement protocols (HTTP, WebSocket, etc.) for real-time communication between clients and the server.

Use encryption and secure protocols (SSL/TLS) to ensure data security and privacy.

3. Implementation

a. Frontend Development:

Develop the user interface based on the designed UI/UX.

Implement features for sending/receiving messages, managing contacts, creating groups, etc.

Ensure responsiveness and compatibility across different devices and browsers.

b. Backend Development:

Develop APIs for user authentication, message handling, and other functionalities.

Implement message storage and retrieval mechanisms.

Set up server-side logic for managing chat sessions, notifications, and user interactions.

c. Database Implementation:

Create database schemas and tables.

Implement data access logic and optimize queries for efficient data retrieval.

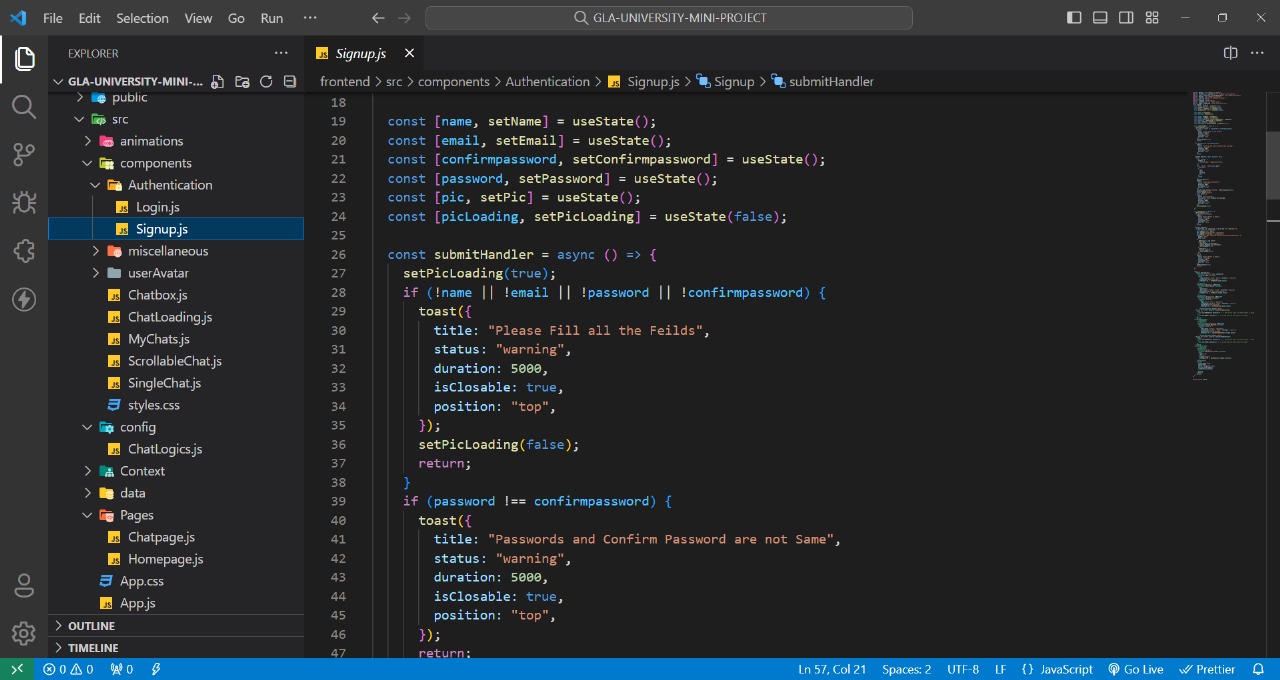
5. User Support and Feedback

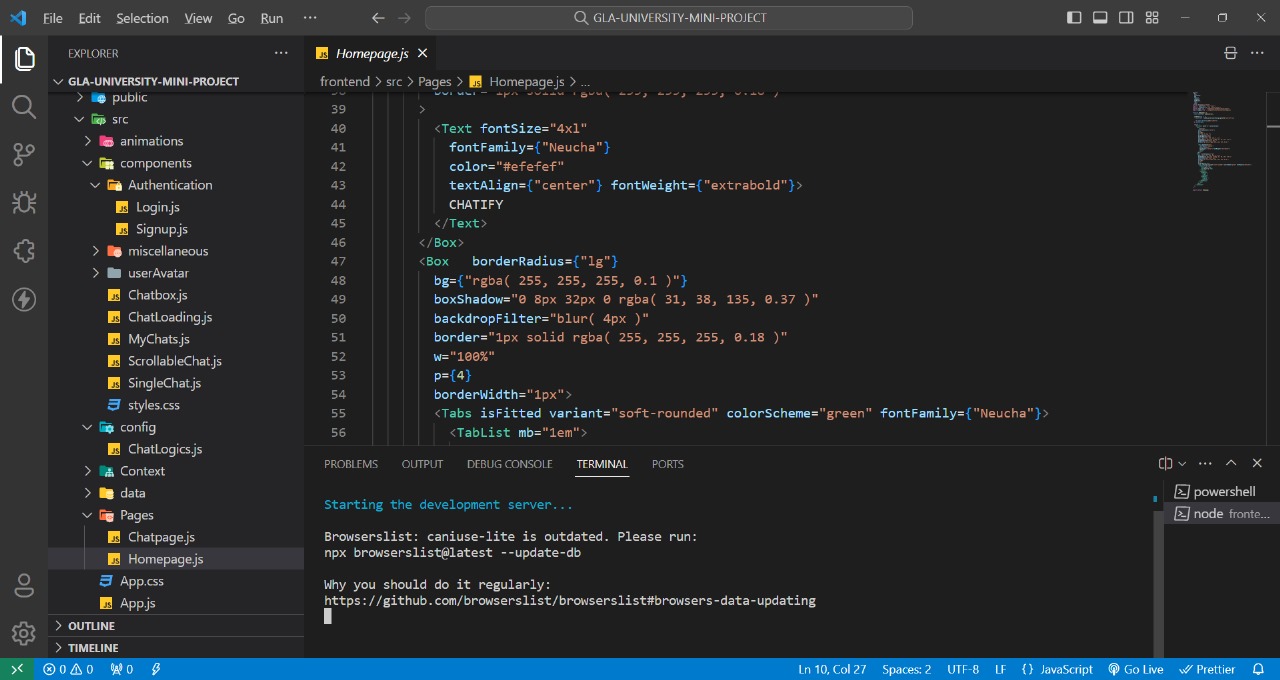
Provide user support channels for feedback, bug reports, and feature requests.

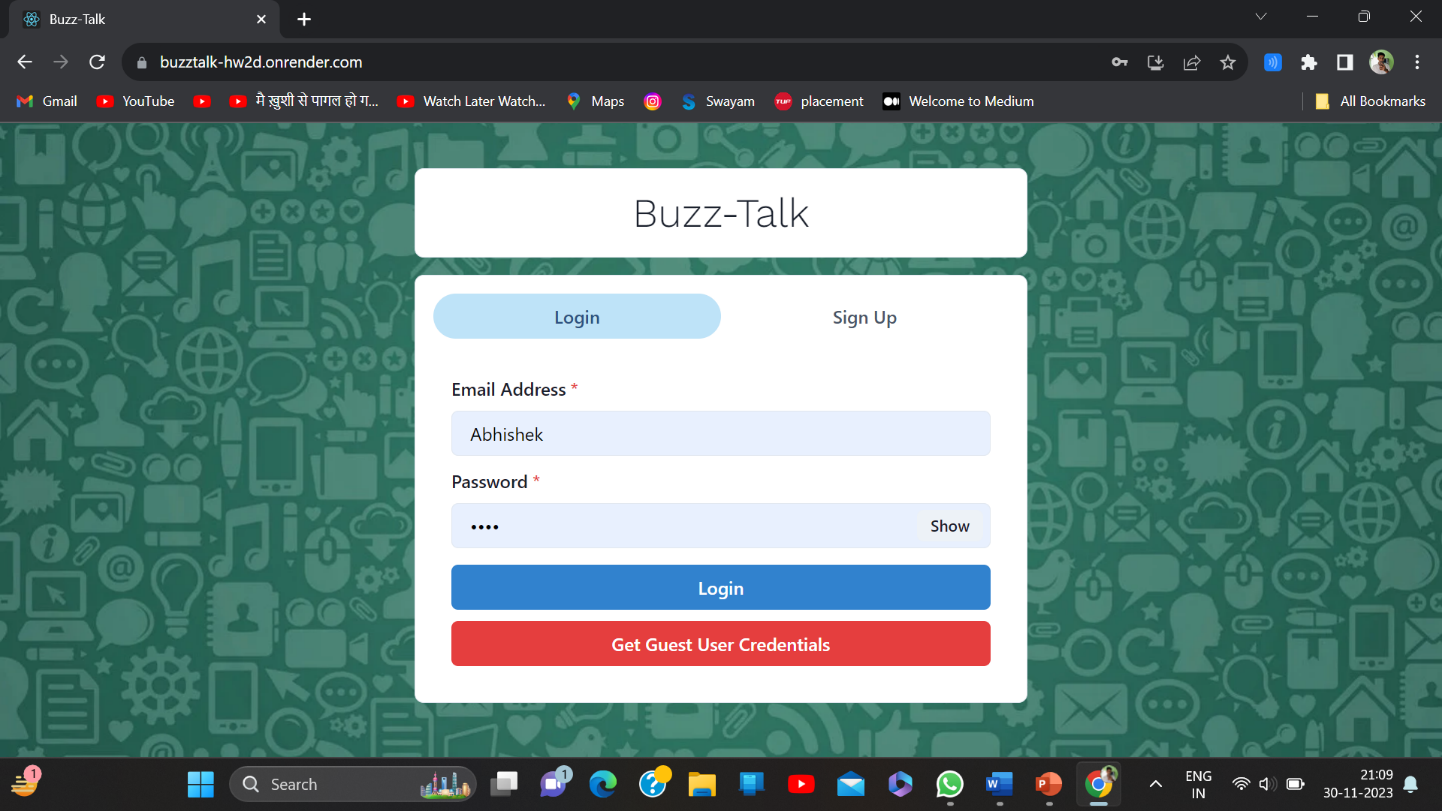
Continuously improve the application based on user suggestions and requirements.

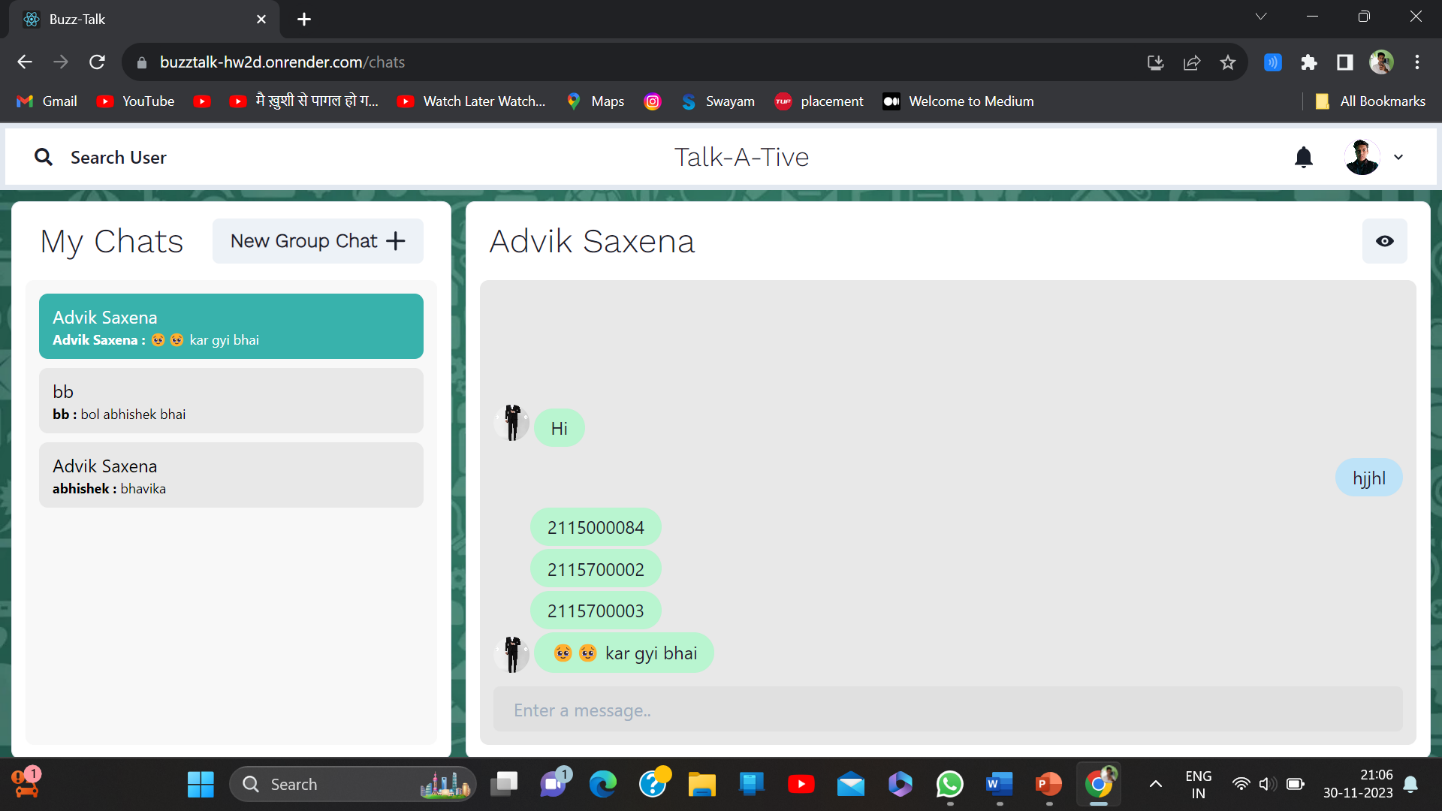
Remember, the design and implementation specifics may vary based on the technology stack, platform, and specific requirements of the chat application. Continuous iteration and improvement are essential to ensure a robust, user-friendly, and secure chat application.

**RESULT**









**CONCLUSION**

In conclusion, the chat application project has successfully delivered a feature-rich and user-friendly platform that revolutionizes communication and file-sharing experiences. With real-time messaging, secure file handling, and comprehensive record-keeping, the application meets the demands of modern collaboration. The emphasis on security, cross-platform compatibility, and a scalable design positions this project as a reliable solution for diverse communication needs. The successful integration of innovative features, coupled with a well-documented codebase, ensures the project's sustainability and adaptability to future advancements in the dynamic landscape of digital communication.

**REFRENCES**

Node js: https://nodejs.org/en/docs

MongoDb: <https://www.mongodb.com/docs/>

Express Js: https://expressjs.com/en/4x/api.html

Socket io: https://socket.io/docs/v4/tutorial/introduction