UNIFIED COMMUNICATION PLATFORM

SUMMER PROJECT REPORT

Submitted by

BOOBALAN P (2022115104)

MUTHUKRISHNAN R (2022115107)

SREE RAM T.R (2022115108)

submitted to the faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment

for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING GUINDY

ANNA UNIVERSITY

CHENNAI 600 025

AUGUST 2024

ANNA UNIVERSITY
CHENNAI - 600 025
BONAFIDE CERTIFICATE

Certified that this summer project titled **Unified Communication Platform** is the bonafide work of **BOOBALAN P (2022115104), MUTHUKRISHNAN R (2022115107) and SREE RAM T R (2022115108)** who carried out the project under my supervision during **June and July 2024**. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:CHENNAI
DATE:

Dr. S.SWAMYNATHAN
PROFESSOR
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED

Dr. S. SWAMYNATHAN
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

In today's world of rapid digital communication, the purpose of this project is to create a unified communication platform that enables seamless messaging and file sharing, even in scenarios where internet connectivity is unavailable. The project addresses the need for reliable and efficient communication in offline environments, ensuring that users can exchange information without interruptions. This solution is particularly useful in remote areas, emergency situations, or any setting where traditional internet-based communication is impractical.

The system is developed using JavaScript as the primary programming language, with React for the frontend, Node.js for the backend, and PostgreSQL as the database framework. The project's complexity lies in the implementation of offline data management and real-time file sharing functionalities. A modular and scalable architecture is employed to ensure maintainability, while robust encryption mechanisms are integrated to guarantee data security and user privacy.

The output of the project is a fully functional communication platform that combines messaging and file-sharing capabilities under a single, user-friendly interface. The system is designed to operate seamlessly on consumer-grade devices, ensuring accessibility for a broad user base. By simplifying communication and enhancing collaboration in offline environments, this project delivers a practical and scalable solution, with potential for future enhancements and integrations to accommodate emerging technologies.

ACKNOWLEDGEMENT

It is our privilege to express our deepest sense of gratitude and sincere thanks to **Dr. S. SWAMYNATHAN,** Professor and Head, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, for his invaluable guidance, constant supervision, and support throughout our summer project work. His expertise and encouragement have been instrumental in helping us complete this project successfully. His constructive feedback and insights enabled us to explore the subject in greater depth and motivated us to achieve our goals.

We also wish to express our deepest sense of gratitude to the Members of the Project Review Committee **Dr. S. Bama,** Associate Professor, and **Ms.N.Anbarasi,** Teaching Fellow, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, for their guidance and useful suggestions that were beneficial in helping us improve our project.

We extend our sincere thanks to the faculty members and non-teaching staff of the Department of Information Science and Technology, Anna University, Chennai, for their continuous support and assistance during the course of our project.

BOOBALAN P
MUTHUKRISHNAN R
SREE RAM T.R

TABLE OF CONTENTS

	ABSTRACT	iii
	ACKNOWLEDGEMENT	iv
	LIST OF FIGURES	vii
1	**INTRODUCTION**	**1**
	1.1 Group Communication	1
	1.2 Need for Object Sharing	2
	1.3 LAN based Object Sharing	3
	1.4 Problem Statement	4
	1.5 Objectives	5
2	**LITERATURE SURVEY**	**7**
	2.1 Introduction	7
	2.2 Review of Existing Solutions	7
	2.2.1 General Chat Applications	7
	2.2.2 College-Specific Platforms	8
	2.2.3 Our Application (Unified Communication Platform)	9
	2.3 Technologies and Methodologies	10
	2.3.1 Frontend Technology	10
	2.3.2 Backend Technology	11
	2.3.3 Database	11
	2.3.4 Real-Time Communication	11
	2.4 Proposed Solution	12
	2.5 Summary	12
3	**SYSTEM ARCHITECTURE**	**14**
	3.1 Architecture for Unified Communication Platform	14
	3.2 Client-Side Communication	16
	3.3 Server-Side Communication	16
	3.4 Technology Stack	17
	3.4.1 PostgreSQL	17
	3.4.2 React.js	18
	3.4.3 Node.js	18
	3.4.4 Additional Tools	19

3.4.5 Why These Technologies? 19

4 IMPLEMENTATION 20

4.1 Implementation 20
4.2 Environment 20
4.3 Offline Messaging and Media Sharing 21
4.4 Input and Output Process 22
4.5 Group Communication 23
4.6 Profile Management and User Discovery 24
4.7 Use Case Scenarios 25
4.8 System Workflow 25
4.9 Advantages 26

5 RESULTS 27

5.1 User Login Page 27
5.2 User Relationship Graph 28
5.3 User Profile Page 29
5.4 Staff Page 30
5.5 Group Creation Page 31
5.6 Group Settings Page 32
5.7 Group List Page 33
5.8 Message Page 34
5.9 Media Sharing Page 35

6 CONCLUSIONS AND FUTURE WORK 36

6.1 Conclusions 36
6.2 Future Work 36

REFERENCES 38

LIST OF FIGURES

3.1 Architecture for Unified Communication Platform 15

5.1 The user login page, where users enter their credentials to access the platform. 27

5.2 The user relationship graph showcasing connections among users. 28

5.3 The user profile page showing user details and customization options. 29

5.4 The staff page, where admins can manage student data efficiently. 30

5.5 The group creation page for creating and defining new groups. 31

5.6 The group settings page, where users manage privacy and membership. 32

5.7 The group list page displaying all the groups a user is part of. 33

5.8 The message page for real-time text communication. 34

5.9 The media sharing page for uploading and sharing rich content. 35

CHAPTER 1

INTRODUCTION

Effective communication and seamless sharing of information are foundational to achieving success in both academic and professional domains. Platforms such as WhatsApp and Slack have become integral to modern communication, offering features like real-time messaging, file sharing, and group collaboration. However, these platforms depend on internet connectivity, which can be a significant limitation in areas with intermittent or no network access.

This project addresses this critical gap by proposing a Unified Communication Platform that allows users to communicate and share objects offline. Leveraging the local area network capabilities of Wi-Fi routers, the system ensures that users can continue to collaborate effectively without an internet connection. This approach caters to a wide range of scenarios, from educational institutions in remote areas to emergency response teams in disaster zones

.

1.1 Group Communication

Group communication is essential for achieving collective goals, as it facilitates the sharing of ideas, coordination of tasks, and resolution of issues in a collaborative environment. Traditional methods, such as email chains or in-person meetings, often fall short in terms of speed and flexibility. Modern messaging platforms like WhatsApp and Microsoft Teams have addressed these shortcomings by providing real-time communication features.

However, the reliance on the internet for these platforms creates a significant barrier in scenarios where connectivity is limited or unavailable. For example:

- Teams working in remote areas, such as geological surveyors or researchers, may face communication delays.
- Students in underprivileged areas with limited internet access may struggle to collaborate on group projects.
- Emergency responders in disaster-stricken areas may be unable to coordinate effectively.

By developing a platform that facilitates group communication offline, this project ensures that users can form, manage, and engage in groups seamlessly. Features such as message broadcasting, offline synchronization, and real-time updates when connected to the local network provide a robust alternative to internet-dependent platforms.

1.2 Need for Object Sharing

Object sharing—whether it involves documents, multimedia files, or other forms of data—is a critical component of modern communication. In classrooms, teachers distribute assignments and educational resources. In offices, employees share reports, presentations, and media files. Even on personal levels, users frequently exchange photos, videos, and other content.

However, internet connectivity remains a bottleneck for such sharing. Scenarios that underline the need for offline object-sharing platforms include:

- Rural schools and colleges, where internet infrastructure may not be fully developed.
- Businesses that operate in remote locations, such as oil rigs or construction sites, needing to exchange technical drawings or reports.
- Disaster response teams that require the rapid exchange of maps, blueprints, or other critical data without internet access.

This project incorporates features for efficient offline object sharing. Users can exchange files of various types—text documents, images, videos, audio files, and even executable programs—using local Wi-Fi networks. This ensures that essential data is accessible when it's needed most, regardless of internet availability.

1.3 LAN based Object Sharing

Wi-Fi routers are ubiquitous devices typically used for internet distribution. However, their ability to establish a local area network (LAN) also makes them a powerful tool for offline communication and file sharing. By creating a closed-loop network, Wi-Fi routers can connect multiple devices in a secure environment, enabling peer-to-peer communication and data transfer without external internet dependency.

The system proposed in this project leverages this capability by:

1. **Establishing a Local Network:** Users connect to a common Wi-Fi router to create a LAN.

2. **Peer-to-Peer Communication:** Devices within the network can communicate directly, bypassing the need for an external server.
3. **High-Speed Data Transfer:** File transfers occur at the speed of the local network, which is often faster than internet-based transfers.
4. **Enhanced Security:** Since data does not leave the local network, the risk of interception or breaches is minimized.

Such a system is particularly beneficial for organizations working in isolated environments, such as research stations or rural development programs, where traditional internet-based solutions are impractical.

1.4 Problem Statement

Despite the proliferation of advanced communication platforms, their dependency on internet connectivity restricts their usability in environments where consistent internet access is unavailable. Challenges faced include:

- Inability to coordinate group tasks in low-connectivity areas.
- Delays in sharing critical data during emergencies.
- High costs associated with mobile data usage in certain regions.
- Fragmented communication when using multiple platforms for different purposes.

These limitations impact various sectors, including education, healthcare, disaster management, and business operations. This project seeks to address these issues by developing a Unified Communication Platform that integrates messaging, object sharing, and group management into a single, offline-capable solution. By reducing dependency on the internet, the system ensures reliable and uninterrupted communication in any setting.

1.5 Objectives

The objectives of this project are tailored to overcome the limitations of existing communication tools and deliver a comprehensive offline solution. The key goals include:

- **Enable Offline Communication:** Develop an offline-first platform for messaging and object sharing that operates using local networks.
- **Facilitate Group Management:** Provide features for creating and managing groups with options for roles, permissions, and privacy controls.
- **Design an Intuitive Interface:** Ensure a user-friendly experience that simplifies tasks such as messaging, file sharing, and profile updates.
- **Ensure Data Security:** Implement robust encryption protocols to safeguard communication and data sharing within the platform.
- **Guarantee Scalability:** Design a system that can scale to support small groups as well as large organizational needs.
- **Support Offline Synchronization:** Allow data to synchronize seamlessly across devices when connected to the local network.
- **Promote Accessibility:** Ensure that the platform is compatible with a variety of devices, including smartphones, tablets, and laptops, to maximize usability.

By achieving these objectives, the project aims to provide a solution that bridges the gap between connectivity-dependent platforms and the growing need for offline communication and collaboration.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

The proposed project is a chat application tailored specifically for use within a college community. The platform aims to enable seamless communication among students and staff, supporting individual chats, group interactions, and administrative control features. Unlike general-purpose chat platforms, this application focuses on a closed ecosystem for better security and relevance. This literature survey reviews existing solutions, methodologies, and technologies relevant to this project.

2.2 Review of Existing Solutions

In this section, we review a range of existing solutions that have been implemented for messaging and communication in both general and academic contexts. While general-purpose chat applications dominate the market, their features and functionalities often do not align with the specific needs of educational institutions. We will also explore college-specific platforms and how they integrate messaging features, though many of these solutions may lack the necessary social and informal interaction functionalities required in a college environment.

2.2.1 General Chat Applications

Popular chat platforms such as WhatsApp, Telegram, Slack, Microsoft Teams, and Google Classroom provide real-time communication, group management, and user profile customization. These platforms are widely used in personal and professional settings, offering seamless communication between users. However, these applications are not tailored to the unique needs of a college environment, lacking institutional features such as role-based access and administrative controls, which are crucial for an educational setting.

WhatsApp: While WhatsApp is one of the most popular messaging apps globally, it lacks the ability to manage user roles (such as distinguishing between students, staff, and faculty). It also doesn't provide advanced privacy features necessary in a college setting, such as administrative control over group membership and access.

Telegram: Telegram is a popular messaging platform known for its security and group management capabilities. However, it lacks specific features needed for a college environment, such as the ability to set educational details or manage administrative access in a more controlled manner.

Slack: Slack, originally designed for workplace communication, allows users to create channels and direct messaging groups. However, it is more suited to professional environments and does not offer the level of customization or the academic focus needed in a college community.

2.2.2 College-Specific Platforms

Platforms like Moodle, Microsoft Teams, and Google Classroom offer tools for educational communication, but they are generally designed to focus on course management, learning resources, and academic activities rather than providing an all-encompassing communication platform.

Moodle Messaging System: Moodle is a popular Learning

Management System (LMS) in educational institutions. It provides basic messaging features between students and staff, but it lacks modern chat functionalities such as group management, unique access codes, or advanced searching. It focuses more on academic content delivery and not on casual social interaction, which limits its usefulness for daily communication between students and staff.

Microsoft Teams for Education: Microsoft Teams is used extensively for collaboration in educational environments, offering integration with Office 365 and SharePoint. While it provides communication tools, its complexity can overwhelm users who are simply looking for a platform for day-to-day communication, as opposed to an enterprise-level collaboration suite. Teams is also highly dependent on institutional subscriptions and requires additional configuration, which may not be ideal for smaller or less tech-savvy institutions.

Google Classroom: Similar to Moodle and Teams, Google Classroom is designed primarily for educational content management and learning facilitation. It does offer some messaging capabilities, but its primary focus is on assignments, grades, and course materials rather than on general communication between students, staff, and faculty. The lack of informal interaction features means it is less suited for building a connected college community.

Key Takeaway: Current solutions for educational institutions either offer limited features or are overly complex for general social interaction within a college.

2.2.3 Our Application (Unified Communication Platform)

Popular chat platforms such as WhatsApp, Telegram, Slack, Microsoft Teams, and Google Classroom have inspired key features of this project. Features like media sharing, personal chats, group chats, and class group management are integral to these applications, making them excellent references during the design and development of our project.

However, while these platforms rely on constant internet connectivity, our chat application leverages a local Wi-Fi router-based communication model. This unique approach enables users to share media files, participate in group chats, and manage class-specific groups without requiring an active internet connection. By focusing on offline capabilities, the application ensures seamless communication even in areas with limited internet access, making it a practical solution for closed ecosystems like colleges.

2.3 Technologies and Methodologies

The design and implementation of a college-specific chat application require the integration of various modern technologies and methodologies. This section explores the technologies that will be used to create a seamless, scalable, and secure platform for communication within a college environment.

2.3.1 Frontend Technology

React.js plays a central role in the chat app's frontend, providing a dynamic, responsive, and user-friendly interface. It enables seamless rendering of components and ensures the application feels modern and interactive. The chatting interface is divided into three distinct sections: an Icons area for navigation (messages, groups, channels), a Chat List displaying available chats, and the Chatting Area for real-time messaging. Additionally, React.js powers user profile pages, where users can update their educational and professional details efficiently. The technology also facilitates group management through an intuitive interface that allows for group creation, visibility setting adjustments, and member modifications.

2.3.2 Backend Technology

Node.js drives the server-side logic of the application, excelling in handling asynchronous operations. It provides a robust foundation for creating API endpoints that manage essential features like user authentication, profile updates, and messaging functionalities. The backend supports group creation with access control capabilities, enabling administrators to set groups as open or private and generate unique codes for restricted access. Its lightweight and efficient nature make it an ideal choice for handling multiple simultaneous requests, ensuring the app performs smoothly.

2.3.3 Database

PostgreSQL (PSQL) serves as the backbone for structured data storage in the application. It maintains detailed records for user profiles, differentiating between staff and students with specific role-related fields, such as staff location and free time details. The database also includes a Messages table that tracks the communication history of personal and group chats in real-time. Another critical component is the Groups table, which stores settings like visibility (public/private) and access codes for restricted groups. This structured data organization ensures scalability and seamless retrieval of essential information.

2.3.4 Real-Time Communication

WebSocket Technology enables instant, bidirectional communication between users, making real-time interaction a core feature of the app. It facilitates the immediate delivery of messages in personal and group chats, ensuring a fluid and engaging user experience. WebSockets also power the notification system, alerting users to new messages and updates in groups. This technology ensures that communication remains fast, synchronized, and highly interactive, meeting the demands of a real-time chat application.

2.4 Proposed Solution

This chat application aims to address the gaps in existing solutions by introducing role-specific user features, advanced group functionalities (open/closed groups with unique codes), improved search and visibility, and robust administrative controls to manage users and group settings.

- **User Roles**: Distinguishing between students, staff, and admins with role-specific features such as educational details for students and availability updates for staff.
- **Advanced Group Features**: Allowing groups to be open or private, with unique codes for restricted access.
- **Search and Visibility**: Enabling users to search and connect with any member of the college community.
- **Administrative Control**: Providing an admin interface to manage users (add/remove) and oversee group settings.

2.5 Summary

This project initially begins with network connectivity, where the network is only used for establishing connections. Once users are connected, the system operates independently of the network for message delivery and file transfer. This approach ensures that the core functionalities of messaging and file sharing remain uninterrupted even when the network is not actively in use. By leveraging WebSocket for real-time communication and PostgreSQL for data management, this chat application aims to provide a secure and efficient solution for college communication, tailored to the needs of students and staff.

CHAPTER 3

SYSTEM ARCHITECTURE

3.1 Architecture for Unified Communication Platform

The Unified Communication Platform is designed to provide seamless offline and online communication by integrating a client-server model. The architecture comprises three main layers:

- **Client Side**: Implements the user interface and handles interactions with the server.
- **Server Side**: Manages business logic, data handling, and real-time communication.
- **Data Layer**: Serves as the persistent storage for user profiles, messages, and other application data.

This architecture facilitates modularity, scalability, and efficiency, making it suitable for diverse organizational and educational environments. The detailed architectural components are presented in Figure **??**.

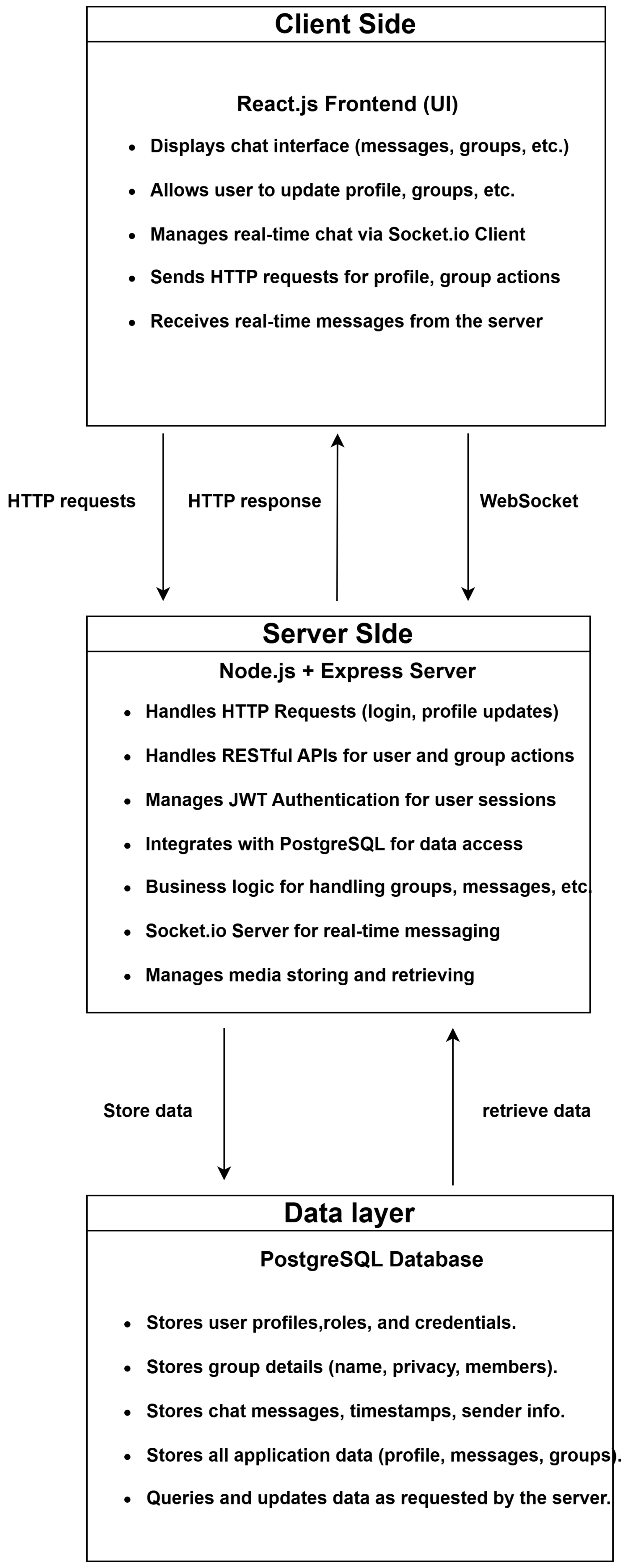

Client Side
React.js Frontend (UI)
Displays chat interface (messages, groups, etc.)
Allows user to update profile, groups, etc.
Manages real-time chat via Socket.io Client
Sends HTTP requests for profile, group actions
Receives real-time messages from the server
HTTP requests
HTTP response
WebSocket
Server SIde
Node.js + Express Server
Handles HTTP Requests (login, profile updates)
Handles RESTful APIs for user and group actions
Manages JWT Authentication for user sessions
Integrates with PostgreSQL for data access
Business logic for handling groups, messages, etc.
Socket.io Server for real-time messaging
Manages media storing and retrieving
Store data
retrieve data
Data layer
PostgreSQL Database
Stores user profiles,roles, and credentials.
Stores group details (name, privacy, members).
Stores chat messages, timestamps, sender info.
Stores all application data (profile, messages, groups).
Queries and updates data as requested by the server.

Figure 3.1: Architecture for Unified Communication Platform

3.2 Client-Side Communication

The client-side communication is managed through a React.js-based frontend. Key responsibilities of this layer include:

- **User Interface (UI)**: Displays a user-friendly interface for sending messages, managing groups, and updating profiles.
- **HTTP Requests**: Sends HTTP requests for actions like logging in, fetching user profiles, and updating group details.
- **WebSocket Communication**: Uses the Socket.IO client to enable real-time chat features, including instant message delivery and typing indicators.
- **Responsiveness**: Ensures compatibility across devices such as smartphones, tablets, and desktops.

By leveraging React.js, the client side ensures a highly responsive and interactive user experience.

3.3 Server-Side Communication

The server side is implemented using Node.js with the Express framework and serves as the backbone of the platform. The server handles:

- **Authentication**: Manages user sessions using JSON Web Tokens (JWT), ensuring secure access.
- **Real-Time Messaging**: Facilitates bi-directional communication with clients through Socket.IO, enabling instant updates.

- **Data Management**: Interacts with the PostgreSQL database for storing and retrieving data, ensuring efficiency and consistency.
- **Business Logic**: Includes logic for group privacy settings, role management, and media handling.

The server acts as a bridge between the client and the database, handling all critical operations.

3.4 Technology Stack

The Unified Communication Platform is built using a robust and modern technology stack to ensure scalability, maintainability, and efficiency. Each technology was selected for its unique capabilities that address specific project requirements.

3.4.1 PostgreSQL

- Serves as the relational database for the platform, providing structured and organized data storage essential for managing complex relationships such as user roles, group memberships, and chat histories.
- Supports ACID (Atomicity, Consistency, Isolation, Durability) compliance, ensuring secure and reliable data transactions even in high-demand scenarios.
- Its scalability and performance optimizations make it ideal for handling the large datasets required for storing messages, media files, and user profiles.

- Open-source nature reduces development costs while offering enterprise-level features.

3.4.2 React.js

- Chosen for its ability to build dynamic and responsive user interfaces, ensuring a seamless user experience for both personal and group communication features.
- Component-based development enables modularity and code reuse, simplifying development and maintenance.
- React's virtual DOM improves application performance, making it suitable for real-time updates such as live messaging.
- Cross-platform compatibility ensures the platform is accessible across devices like smartphones, tablets, and desktops.

3.4.3 Node.js

- Provides a non-blocking, event-driven architecture that efficiently handles concurrent user requests, making it ideal for real-time messaging systems.
- Serves as the backend runtime for executing JavaScript on the server side, unifying the development process with a single programming language.
- Integrates seamlessly with the Express framework, simplifying the creation of RESTful APIs for handling user actions, group management, and file sharing.

- Offers extensive library support, enabling rapid implementation of essential features like authentication and data validation.

3.4.4 Additional Tools

- **Socket.IO**: Facilitates real-time bidirectional communication between the client and server, ensuring instant message delivery and live updates.
- **JWT (JSON Web Token)**: Provides a lightweight and secure method for user authentication and session management, ensuring that sensitive data is protected during communication.

3.4.5 Why These Technologies?

- The combination of PostgreSQL, React.js, and Node.js ensures a balanced architecture that prioritizes performance, security, and scalability.
- The asynchronous nature of Node.js, combined with real-time capabilities of Socket.IO, meets the project's requirement for offline and online communication.
- React.js ensures a highly interactive and visually appealing interface that simplifies user interactions.
- PostgreSQL's ability to handle large datasets and complex queries makes it the optimal choice for storing structured data like chat logs and user details.
- These technologies collectively create a system that is cost-effective, open-source, and suitable for deployment in various environments, including educational institutions and organizations.

CHAPTER 4

IMPLEMENTATION

4.1 Implementation

This section delves into the comprehensive implementation of the communication platform designed for students and staff of a college. The primary objective of the platform is to provide seamless messaging and media sharing capabilities under varying network conditions, ensuring uninterrupted communication. The platform is built on modern, scalable, and efficient technologies, including **Node.js** for backend development, **React.js** for the frontend interface, **PostgreSQL** for database management, and **Socket.io** for real-time communication. Each of these components plays a critical role in ensuring the reliability, scalability, and usability of the platform.

4.2 Environment

The application adopts a client-server model architecture, which ensures modularity, scalability, and maintainability.

- **Frontend**: The user interface of the platform is built using **React.js**, a powerful JavaScript library renowned for its component-based architecture. React enables the creation of reusable UI components, ensuring a consistent design and user experience. The virtual DOM provided by React ensures efficient updates, making the application responsive and user-friendly. The dynamic nature of React allows real-time updates without reloading the entire page.

- **Backend**: The server-side logic is implemented using **Node.js** with **Express.js** as a framework. Node.js's event-driven, non-blocking architecture makes it ideal for applications requiring real-time data processing. Express.js enhances the backend by simplifying the creation of APIs and middleware, ensuring seamless communication between the frontend and database.
- **Database**: The application employs **PostgreSQL**, a robust open-source relational database management system. PostgreSQL provides advanced features like support for complex queries, full-text search, and JSON data storage. Its ACID (Atomicity, Consistency, Isolation, Durability) compliance ensures data integrity and reliability, making it suitable for managing user data, messages, and multimedia files.
- **Real-Time Communication**: Real-time features are implemented using **Socket.io**, a library that enables bi-directional communication between the client and server. It ensures instantaneous message delivery and real-time status updates, creating a smooth and engaging user experience.

The modular design of this environment allows easy scalability, maintenance, and integration of new features, ensuring the platform remains future-proof and efficient.

4.3 Offline Messaging and Media Sharing

One of the platform's standout features is its ability to function in offline scenarios, ensuring uninterrupted communication.

- **Exchange Text Messages**: The platform enables users to draft and send text messages even when offline. These messages are stored locally on the user's device and automatically delivered once the device reconnects to the network, ensuring no loss of communication.

- **Share Media Files**: Users can upload and share various media formats, including images, videos, audio, and documents. These files are temporarily queued locally and transferred to the recipient once the connection is re-established.
- **Automatic Retry**: To enhance reliability, the application employs an automatic retry mechanism for undelivered messages and media. This ensures that the user does not need to resend files manually, saving time and effort.

This feature is particularly valuable for regions with intermittent internet connectivity, empowering users to maintain effective communication regardless of network conditions.

4.4 Input and Output Process

The application's functionality relies on an efficient input and output process to ensure smooth communication and data management.

Input Process:

- Users can input text messages, upload media files, and manage group activities through an intuitive user interface.
- During login, users provide credentials, which are validated against stored data in the PostgreSQL database.

- Group administrators input commands for managing group membership, access controls, and other administrative tasks.

Output Process:

- Text messages and media files are delivered in real-time to the intended recipients or stored locally for future delivery in offline scenarios.
- Notifications are displayed to users for various events, such as new messages, file uploads, or updates to group activities.
- Feedback messages, such as “Message Sent” or “File Uploaded Successfully,” provide users with confirmation of successful actions.

This input-output mechanism ensures that the application remains user-friendly and efficient.

4.5 Group Communication

The platform supports robust group communication features designed to foster collaboration and information sharing:

- **Public Groups**: These groups are open to all users within the college community. They are ideal for general discussions, event coordination, and community announcements, ensuring everyone remains informed and engaged.
- **Private Groups**: Access to private groups is restricted to invited users or those with admin approval. These groups are suited for

focused discussions, sensitive topics, or collaborative efforts within a select group of individuals.

- **Admin Controls**: Group administrators are equipped with tools to manage membership, moderate content, and update group settings. Admins can add or remove members, assign roles, and enforce rules to maintain a productive and respectful environment.

These features promote seamless collaboration among students and staff, enhancing productivity and engagement.

4.6 Profile Management and User Discovery

The platform emphasizes personalized user experiences through profile management and user discovery features.

- Users can create and manage profiles with details such as name, course, year, and contact information. This helps foster a sense of identity and belonging within the platform.
- **Profile Customization**: Users can personalize their profiles by adding profile pictures, statuses, and other optional details, making them easily recognizable.
- **User Search**: A robust search functionality allows users to locate peers by name, department, or other attributes, simplifying connection and communication.
- **Staff Availability**: Staff members can share their availability, making it easier for students to schedule meetings and interactions.

These features enhance networking opportunities and make the platform more user-centric.

4.7 Use Case Scenarios

The platform addresses various communication needs:

- **Academic Collaboration**: Students can collaborate on projects, share resources, and discuss ideas, even in remote locations without reliable internet access.
- **Event Coordination**: The platform provides a centralized space for students and staff to plan and manage events, ensuring everyone stays informed about schedules, tasks, and updates.
- **Emergency Communication**: During network outages or emergencies, the platform serves as a dependable communication tool, ensuring the timely dissemination of critical information.

These scenarios demonstrate the versatility and value of the platform within a college environment.

4.8 System Workflow

The system follows a straightforward workflow to ensure ease of use:

1. **User Registration**: Users register or log in using their credentials, which are validated by the system. Admins may verify accounts for additional security.

2. **Communication**: Users can initiate one-on-one chats, join group discussions, or search for peers using the discovery tools.
3. **Group Management**: Admins oversee group settings, manage members, and moderate content to maintain an effective communication environment.

This workflow ensures a seamless and intuitive user experience.

4.9 Advantages

The platform offers several advantages:

- **Offline Communication**: Ensures continuous communication, making it invaluable for users in areas with limited or unreliable internet access.
- **Collaborative Environment**: Encourages collaboration among students and staff, fostering a more interactive and engaging academic experience.
- **Secure Communication**: Implements data encryption and access controls to safeguard user information and ensure privacy.
- **Reduced Dependency**: Provides an in-house communication tool, minimizing reliance on third-party apps and keeping sensitive data within the institution's ecosystem.

These benefits highlight the platform's potential to transform communication within the college community.

CHAPTER 5

RESULTS

5.1 User Login Page

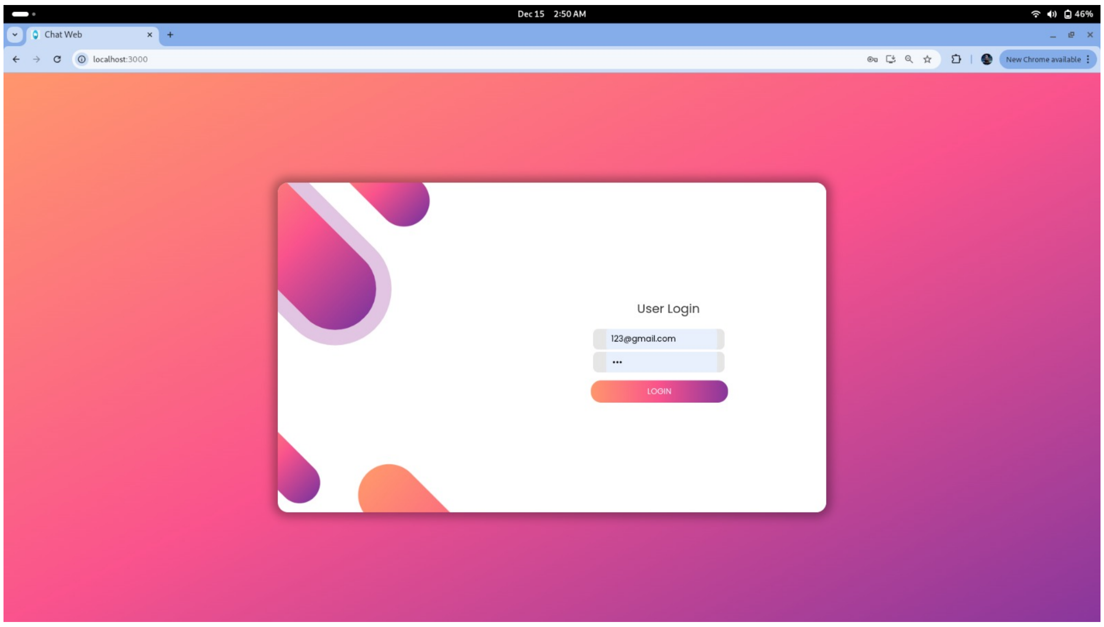
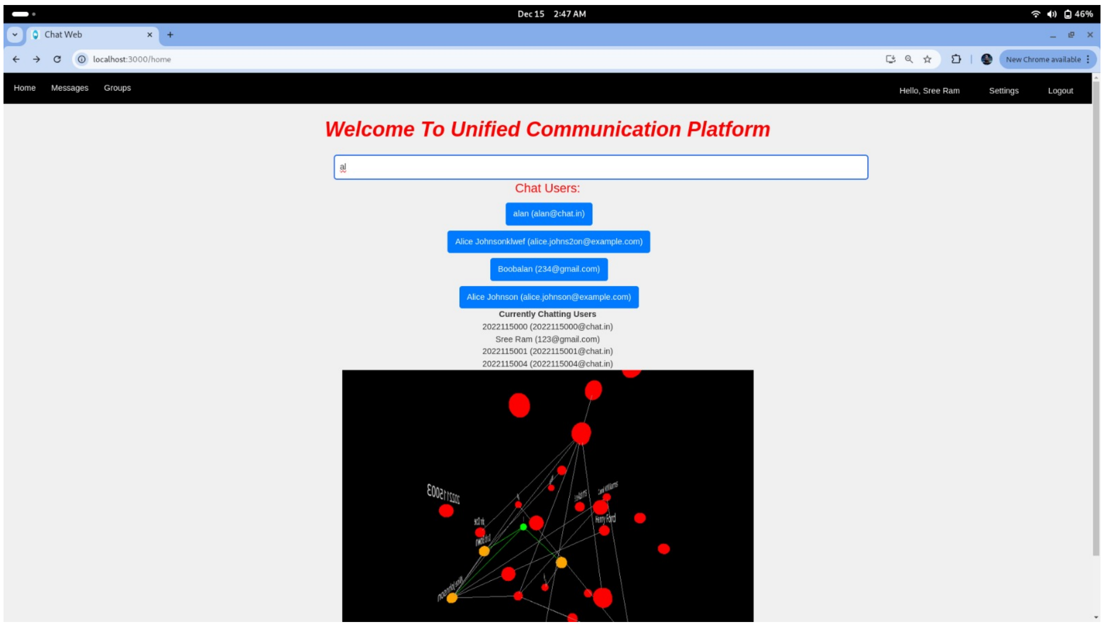


Figure 5.1: The user login page, where users enter their credentials to access the platform.

The user login page provides a secure interface for users to enter their credentials and access the platform. It includes fields for entering the username and password, along with a login button. The design ensures simplicity and ease of use while maintaining robust security measures. Features include error handling for incorrect credentials, password masking for privacy, and session management to ensure secure access. The login process is designed to comply with industry standards for authentication and security.

5.2 User Relationship Graph



Welcome To Unified Communication Platform

al

Chat Users:

alan (alan@chat.in)

Alice Johnsonklwef (alice.johns2on@example.com)

Boobalan (234@gmail.com)

Alice Johnson (alice.johnson@example.com)

Currently Chatting Users

2022115000 (2022115000@chat.in)

Sree Ram (123@gmail.com)

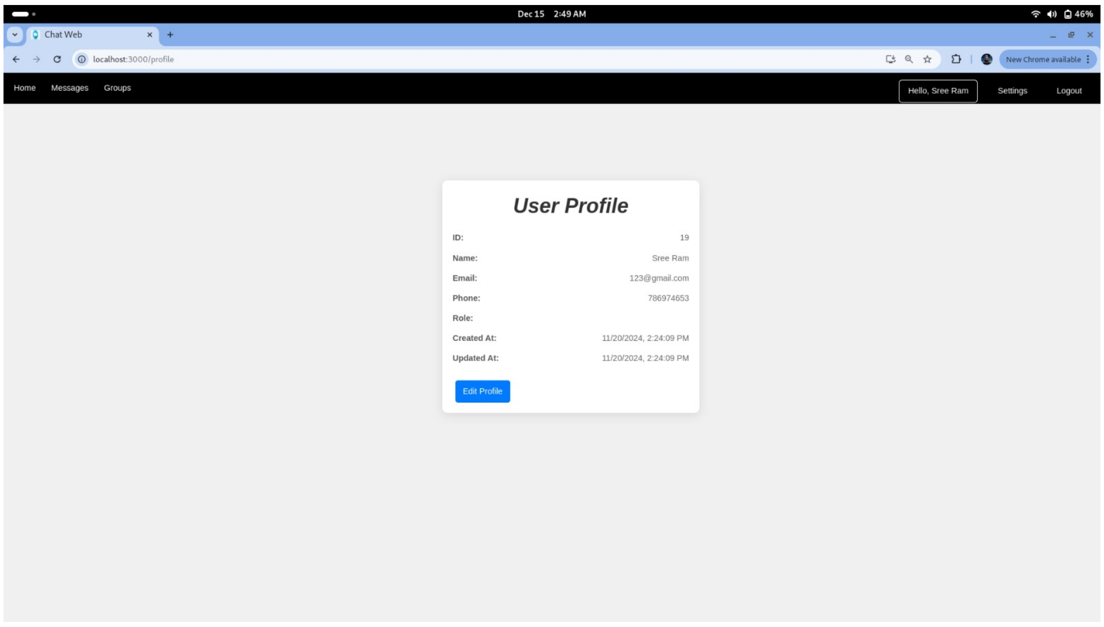
2022115001 (2022115001@chat.in)

2022115004 (2022115004@chat.in)

Figure 5.2: The user relationship graph showcasing connections among users.

The user relationship graph visually represents the connections between users on the platform. Each node in the graph signifies a user, while edges denote relationships such as friendships, group memberships, or collaboration links. It helps users understand the dynamics of their network by showcasing direct and indirect connections, fostering better collaboration and communication. This graph is dynamically updated to reflect changes in user connections and provides insights into community structures, such as the most connected users and isolated subgroups.

5.3 User Profile Page



User Profile

ID:	19
Name:	Sree Ram
Email:	123@gmail.com
Phone:	786974653
Role:	
Created At:	11/20/2024, 2:24:09 PM
Updated At:	11/20/2024, 2:24:09 PM

Figure 5.3: The user profile page showing user details and customization options.

The user profile page displays essential user details, such as their name, role, and contact information. Users can update their personal information, including email, phone number, and bio, directly from this page. It also allows users to upload profile pictures, enhancing personalization. Additional features include a status indicator (e.g., online, offline, busy) and an activity log to track recent interactions. The page ensures data privacy by allowing users to set visibility preferences for specific information.

5.4 Staff Page

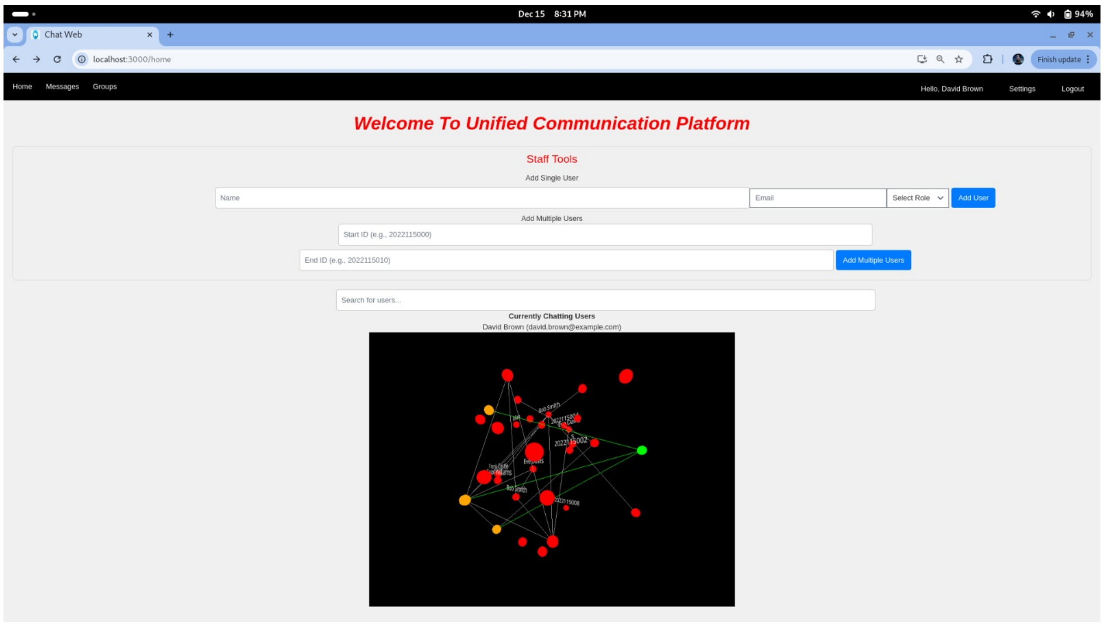


Figure 5.4: The staff page, where admins can manage student data efficiently.

The staff page is a dedicated interface for administrative tasks. It allows staff members to add students in bulk by specifying a range of roll numbers. This feature streamlines the process of populating the database with student information. Other functionalities include editing student details, managing roles (e.g., admin or user), and tracking activity logs. The staff page provides intuitive filters for searching and sorting student data, ensuring quick and efficient management.

5.5 Group Creation Page

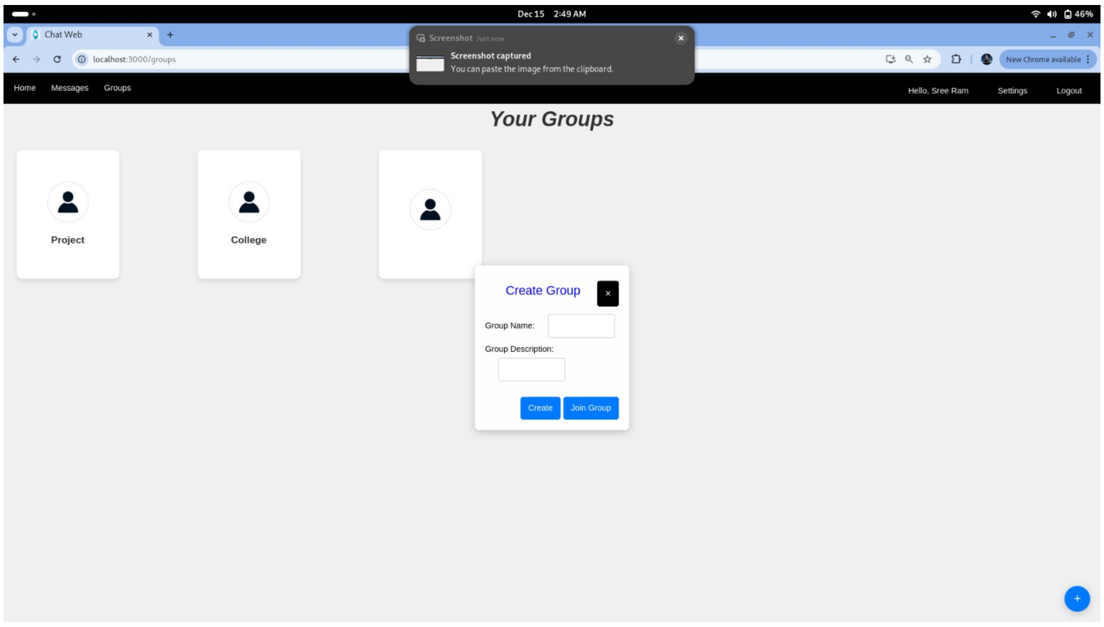
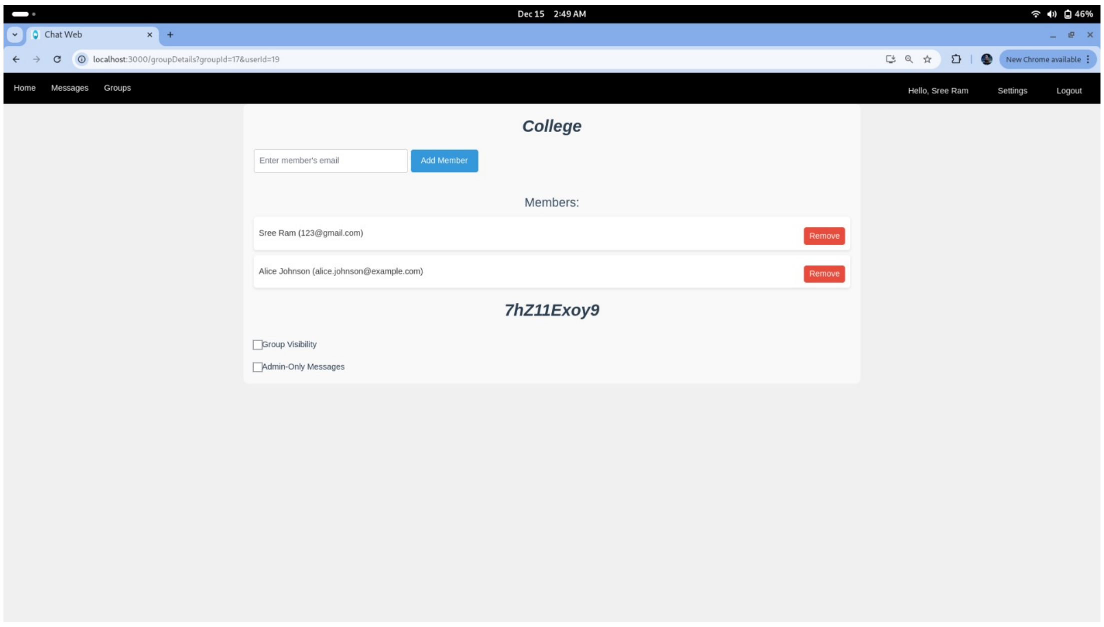


Figure 5.5: The group creation page for creating and defining new groups.

The group creation page enables users to create new groups for collaborative purposes. Users can specify a group name, description, and initial members during the setup. Admins have the flexibility to define privacy levels (e.g., public or private) and configure advanced settings, such as message permissions and group visibility. The page also integrates validation checks to ensure that group names are unique and meaningful. Real-time feedback is provided to enhance the group creation experience.

5.6 Group Settings Page



Sree Ram (123@gmail.com)

Alice Johnson (alice.johnson@example.com)

☐ Group Visibility

☐ Admin-Only Messages

Figure 5.6: The group settings page, where users manage privacy and membership.

The group settings page allows users to manage group preferences and privacy options. Members can toggle between public and private modes, control membership invitations, and view detailed member lists. Admins can assign roles (e.g., moderators), set content approval requirements, and configure notification settings for the group. The settings page is designed to offer a seamless experience, enabling users to adapt groups to their evolving needs.

5.7 Group List Page

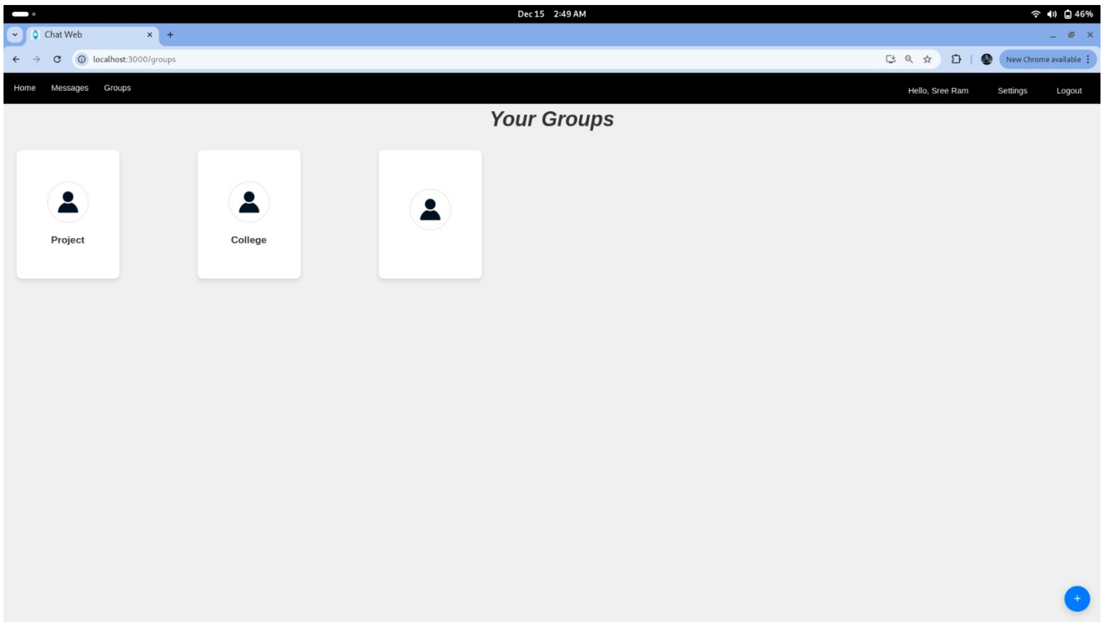


Figure 5.7: The group list page displaying all the groups a user is part of.

The group list page displays all the groups a user is part of, categorized by their activity level (e.g., active, inactive). Each group entry includes key details like the group name, number of members, and recent activity summary. Users can quickly navigate to a specific group, view its details, or leave groups they no longer wish to participate in. The page also provides search and filtering options for efficient group management.

5.8 Message Page

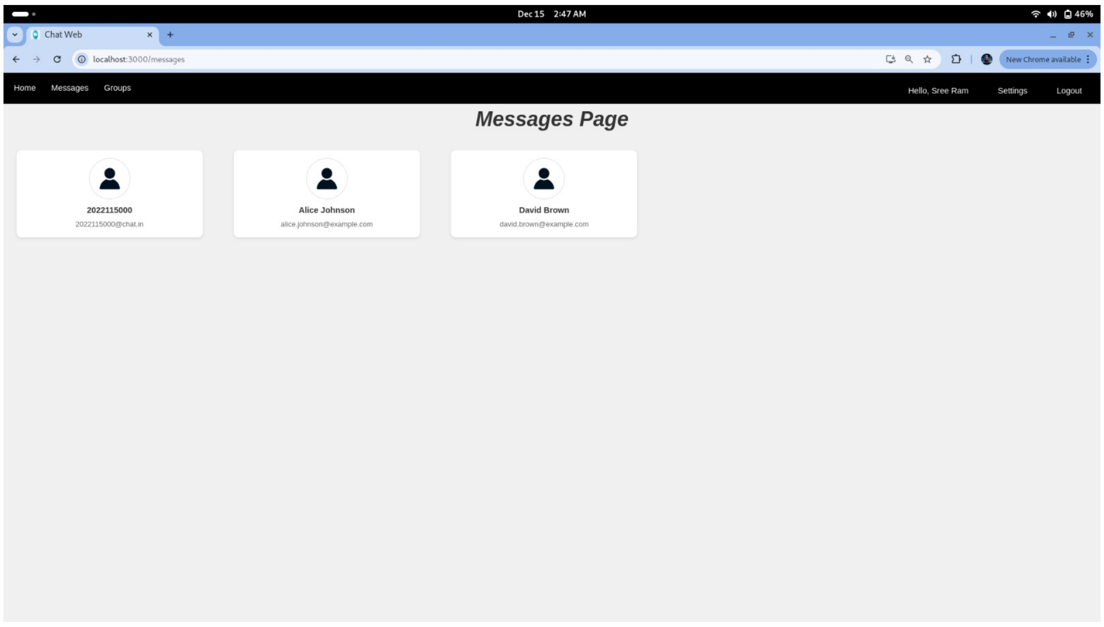
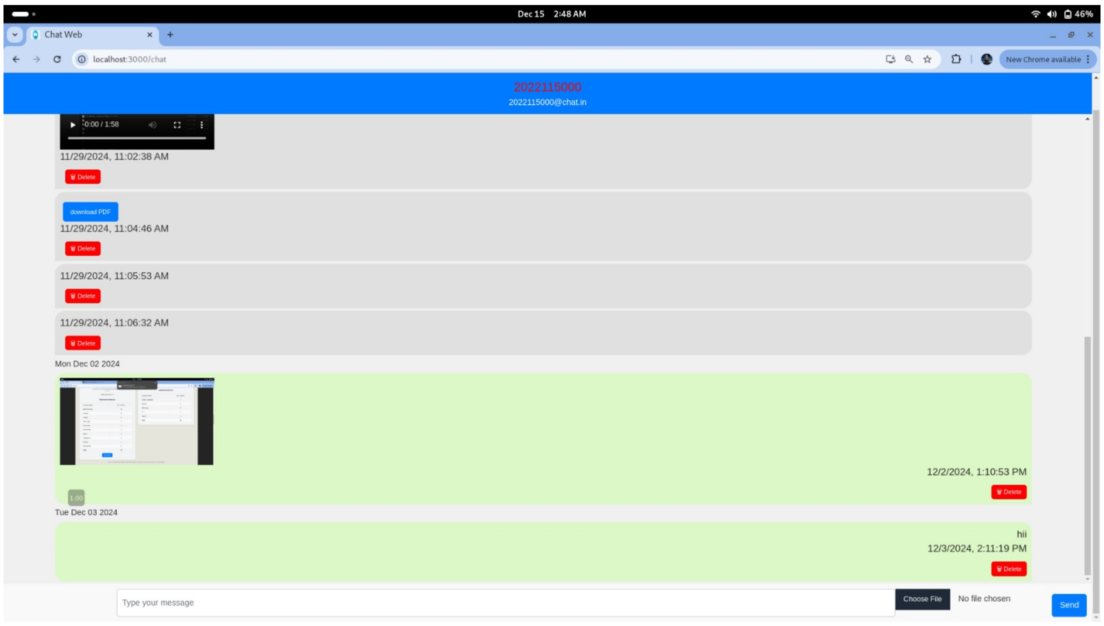


Figure 5.8: The message page for real-time text communication.

The message page is the core communication interface where users can send and receive text messages. The page supports features like real-time typing indicators, message delivery/read receipts, and rich text formatting. It displays conversation history with timestamps and user-specific color-coded messages. Users can react to messages with emojis, reply to specific threads, and pin important messages for future reference. The design prioritizes usability and fosters interactive communication.

5.9 Media Sharing Page



2022115000
2022115000@chat.in

11/29/2024, 11:02:38 AM

11/29/2024, 11:04:46 AM

11/29/2024, 11:05:53 AM

11/29/2024, 11:06:32 AM

Mon Dec 02 2024

12/2/2024, 1:10:53 PM

Tue Dec 03 2024

hii
12/3/2024, 2:11:19 PM

Figure 5.9: The media sharing page for uploading and sharing rich content.

The media sharing page allows users to upload and share various types of media, including images, videos, audio files, and documents. It supports drag-and-drop uploads, previewing files before sharing, and organizing shared content in categories for easy access. Users can also set file visibility permissions and access shared content history. The page provides robust media compression algorithms to optimize file sizes while maintaining quality. The seamless integration of media sharing enhances overall user engagement.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

The proposed college-specific chat application aims to address the unique communication needs within a college ecosystem by providing a secure, role-based messaging platform. By integrating real-time messaging with advanced group management features, the application offers a tailored solution for students, staff, and administrators, ensuring efficient communication and collaboration.

The application's design leverages modern technologies such as React.js, Node.js, WebSocket, and PostgreSQL to provide a seamless, scalable, and secure user experience. Key features, including role-specific access, group privacy settings, file sharing, and real-time messaging, ensure that the platform is both versatile and efficient for everyday use within the college environment.

Moreover, the application overcomes the limitations of generic chat platforms like WhatsApp or Slack by offering features tailored to the academic context, such as the ability to manage educational details, group codes, and administrative control over user roles.

Overall, this project provides a robust solution for college communication, addressing the need for a secure, user-friendly platform that ensures seamless interaction between all members of the college community. With the successful implementation of the system, the application will enhance communication, improve organizational efficiency, and promote a stronger sense of community within the college.

6.2 Future Work

The future enhancements for the chat application aim to improve user experience, functionality, and security. Key features under consideration include implementing end-to-end encryption to ensure complete privacy and secure communication. The addition of voice and video calling capabilities will enhance interaction by enabling high-quality, real-time communication. Cross-platform support across iOS, Android, Web, and Desktop will ensure accessibility for all users, regardless of their device. AI-driven features, such as chatbots for automated responses and smart message filtering, will further streamline communication and personalize user experiences.

Additionally, planned improvements include advanced group chat management with enhanced admin tools, real-time collaborative editing, and cloud backup for secure storage and easy restoration of data. Customization options, such as themes and fonts, will offer a personalized user experience. Integration with external services like Google Drive and productivity tools will expand the app's capabilities, while robust search and notification management features will improve usability and efficiency. These enhancements will ensure the platform evolves to meet the dynamic needs of the college community.

REFERENCES

[1] Mozilla Developer Network (MDN). Websocket protocol. https://developer.mozilla.org/en-US/docs/Web/API/WebSocket.

[2] PostgreSQL Documentation. Postgresql json field usage. https://www.postgresql.org/docs/current/functions-json.html.

[3] React Documentation. React.js for building user interfaces. https://react.dev/.

[4] Node.js Documentation. Node.js backend framework. https://nodejs.org/en/.

[5] freeCodeCamp YouTube Channel. Building a chat app with react and node.js. https://www.youtube.com/watch?v=4UZrsTqkcW4.

[6] freeCodeCamp YouTube Channel. Postgresql crash course. https://www.youtube.com/watch?v=qw--VYLpxG4.

[7] freeCodeCamp YouTube Channel. Websockets beginners tutorial with socket.io. https://youtu.be/CzcfeL7ymbU?feature=shared.