WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

CLI (Command Line Interface) Based Chat Tool

Ms. Harshini Lakshmi RK[1], Mrs. Kanimozhi A[2], MSc MPhil Ph.D

[1]UG Student, Department of Computer Science, Sri Krishna Adithya College of Arts and Science, Coimbatore.
[2]Assistant Professor, Department of Computer Science, , Sri Krishna Adithya College of Arts and Science, Coimbatore

ABSTRACT:

Efficient communication systems are essential for seamless interaction in personal, professional, and educational settings. This paper presents the development of a lightweight Command-Line Interface (CLI)-based chat tool that facilitates realtime communication using a client-server architecture. Designed for environments with limited resources, the system provides features such as user authentication, private messaging, group chats, offline message queuing, and optional encryption for secure data exchange. The tool employs socket programming for real-time message delivery and offers scalability to support multiple users simultaneously. By focusing on simplicity and efficiency, this CLI-based chat tool serves as a practical solution for developers, system administrators, and users requiring a reliable communication platform in resource-constrained or terminal-based environments.

1. INTRODUCTION:

In today's digital age, communication plays a pivotal role in personal and professional environments. While many messaging applications exist, a lightweight and efficient solution is often necessary for specific use cases such as server administration, collaborative coding, or environments with limited resources. The CommandLine Interface (CLI)-based chat tool addresses this need by providing a simple yet powerful platform for realtime communication directly within the terminal.

This project leverages a client-server architecture to enable secure and scalable messaging. Users can log in, join chat rooms, send private messages, or participate in group chats, all through a straightforward text-based interface. With features like offline message queuing, chat history storage, and optional encryption, the tool is designed to be reliable and adaptable. By focusing on functionality over complexity, this CLI-based chat application offers an accessible and efficient communication solution for developers, system administrators, and users in low-bandwidth environments.

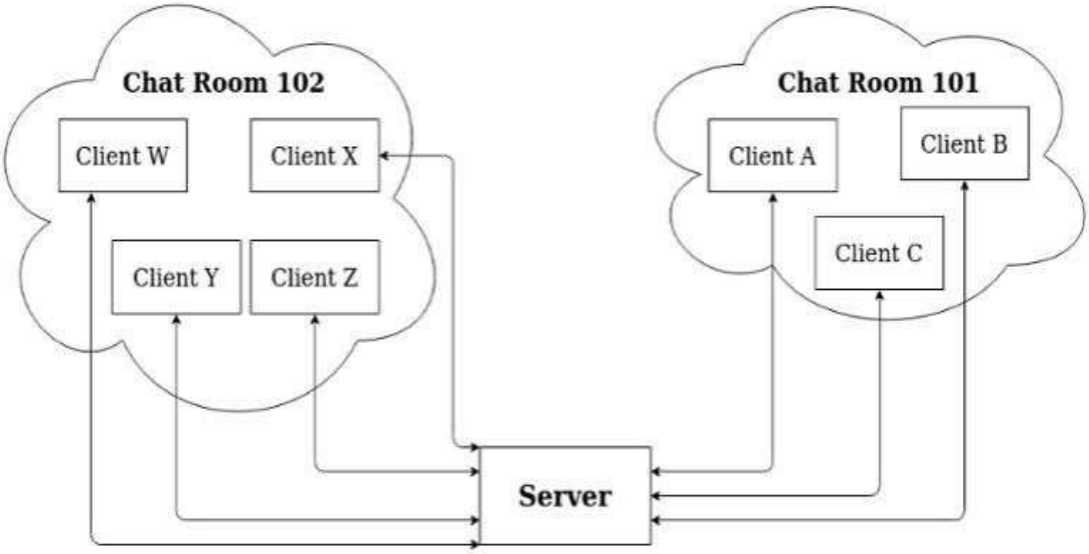
The CLI-based chat tool is a lightweight, real-time messaging application designed for use in a terminal environment. It employs a client-server architecture, where users can log in, join chat rooms, and exchange messages securely. The server handles message routing, user authentication, and optional features like offline message queuing and chat history logging. Built with simplicity and extensibility in mind, the tool can support private chats, group conversations, and encrypted communication for enhanced security. Ideal for lightweight deployments, it serves as a reliable communication solution for personal, educational, or professional use.

2. PROPOSED METHOD:

Command-Line Interface (CLI)-based chat tool designed to provide a lightweight, efficient, and reliable communication platform for real-time messaging. It adopts a client-server architecture where the server manages user connections, message routing, and data storage, while the client offers a simple terminal-based interface for interaction. The system includes features such as secure user authentication, real-time private and group messaging, and dynamic chat room management. It supports offline messaging by storing messages for users who are

International Journal of Research Publication and Reviews, Vol (5), Issue (4), April (2024), Page – 420-426

temporarily disconnected, ensuring they are delivered upon reconnection. Additionally, the system maintains message logs for future reference and offers optional end-to-end encryption for secure communication. Designed to be scalable, it can handle multiple users and large chat rooms simultaneously. The lightweight design ensures compatibility with low-resource environments, while its cross-platform functionality makes it operable on Windows, macOS, and Linux. The system is extensible, allowing for the integration of additional features like file sharing, AI moderation, or web interfaces. This makes it an ideal solution for developers, system administrators, and users in terminal-based or resource-constrained settings.



3. IMPLEMENTATION :

The Command Line Interface (CLI) Chat Tool is implemented using a socket-based architecture to enable real-time communication between multiple clients via a server. Below is a detailed description of the key components and their interactions:

1. Server Initialization:

The server creates a socket, binds it to a specific port, and begins listening for incoming client connections.

2. Client Connec}on Handling:

When a client connects, a new thread is created for the client. This ensures that the server can handle mul}ple clients simultaneously without blocking.

3. Message Broadcas}ng:

The server con}nuously waits for messages from connected clients. When a message is received, it is broadcasted to all other connected clients using e 旳케cient socket communica}on.

4. Client Disconnec}on:

If a client disconnects, the server removes the client from the ac}ve client list, closes its socket, and waits for new connec}ons.

Figure 1.2 Startup / Login prompt

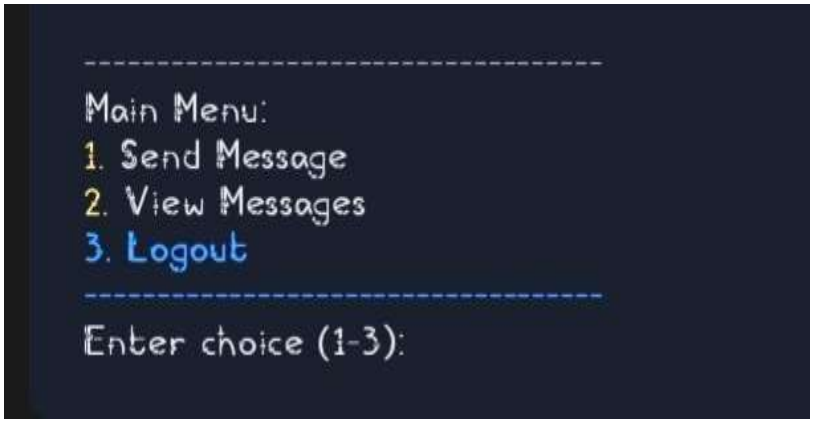


Figure 1.3 Main Menu after login

Figure 1.4 Send Message

Figure1.5 View Message

Figure 1.5 Error Handling

Figure1.6 Logout

4. Applications

1. Internal Communication for Small Teams: A lightweight messaging tool for teams or small businesses to maintain communication without the overhead of full-featured platforms like Slack or Microsoft Teams.
2. Real-Time Customer Support: A live chat tool where customer service agents can communicate with users for troubleshooting or answering queries.
3. Remote Collaboration for Developer: A simple CLI tool for real-time communication between developers working remotely or in different locations.
4. Security and Privacy-Focused Messaging: A private messaging system where messages are encrypted, providing secure communication for sensitive discussions.
5. Educational Collaboration and Group Discussion: A messaging platform for students, educators, or learners to collaborate in study groups, discuss topics, and share resource
6. Incident Reporting and Collaboration in Critical Environment: Used in environments like hospitals, research labs, or emergency response teams for quick reporting and communication during critical situations.
7. Community or Support Forums: A CLI-based chat system used by online communities to facilitate discussions, knowledge sharing, and Q&A sessions.

5. CONCLUSION :

The CLI-based chat tool o旳旳ers a lightweight, e旳케cient, and e旳旳ec}ve solu}on for real-}me communica}on across various domains. Its simplicity makes it highly suitable for small teams, remote collabora}ons, and environments with limited resources or low bandwidth. The tool's ability to facilitate secure, text-based communica}on can be par}cularly bene旳椀cial in privacy-sensi}ve applica}ons, such as secure messaging and customer support.

While the chat tool provides a fundamental approach to messaging, its 旳甦exibility allows for easy customiza}on and extension with features like encryp}on, private messaging, and logging. This makes it adaptable for various use cases, from team communica}on and IT monitoring to educa}onal collabora}on and remote assistance.

Ul}mately, the CLI-based chat tool proves to be an invaluable resource for small businesses, educa}onal ins}tu}ons, developers, and organiza}ons seeking a straighvorward communica}on system with minimal overhead. By o旳旳ering essen}al features with a focus on simplicity and e旳캐ciency, it ensures users can stay connected and produc}ve with ease.

6. Acknowledgement:

I would like to express my hearvelt gra}tude to everyone who contributed to the successful implementa}on of this CLI Chat Tool project. I extend my sincere thanks to [Mentor's /Instructor] for their guidance, invaluable feedback, and encouragement throughout the project. Special apprecia}on goes to my team members for their collabora}on, dedica}on, and problem-solving e旳旳orts. I am also grateful for the availability of online resources and documenta}on that greatly supported the technical aspects of this project. Lastly, I thank my family and friends for their unwavering support and mo}va}on, enabling me to focus and achieve the desired outcomes. This project has been a signi旳椀cant learning experience, allowing me to deepen my understanding of socket programming and real-}me communica}on systems.

7.REFERENCES:

1. "Python Programming: An Introduction to Computer Science" by John Zelle

- This book provides a solid foundation in Python programming, which is essential for building a CLI-based tool using Python.

2. "The Python Standard Library" by Fredrik Lundh

- This reference provides an in-depth overview of Python's standard libraries, including the socket module, which is used for network communica}on in this project.

3. **"Python Networking" by Brandon Rhodes and John Goerzen**

- This book covers the fundamentals of networking in Python, including socket programming and multithreading, which are crucial for building a chat tool.

4. **Python Official Documentation - Socket Programming** https://docs.python.org/3/library/socket.html

- The official documentation for Python's socket library is an essential reference for understanding socket programming in Python.

5. **"A Study on Socket Programming and its Use in Distributed Systems"**

- This paper provides insights into how socket programming is used in distributed systems, which is applicable to building network-based applications like a chat tool.

6. **"Designing a Simple Real-Time Messaging System"**

- This article discusses the basic principles behind real-time communication systems, which can be directly applied to the design of a CLI-based chat tool.

7. **Python 3.x** https://www.python.org/

- The official website for Python, where you can download the latest version and access extensive documentation. 8. **PyCharm**
- https://www.jetbrains.com/pycharm/
- A powerful Integrated Development Environment (IDE) for Python, useful for developing and debugging the chat tool.

9. **Real-Time Chat System Design**

- This online tutorial explores the principles of designing real-time chat systems, useful for understanding scalability and performance considerations in multi-client communication.

These references cover essential resources, documentation, and learning materials that would assist in understanding and building a robust CLI-based chat tool.