INTERNSHIP PROJECT-1 DOCUMENT

ON “Chat Application (JAVA)”

Submitted by:

Navpreet Singh (INTERN)

Submitted To:-

Kanduri Abhinay (FOUNDER)

RITHIN VERMA (CTO)

INDEX

INTRODUCTION:-

The Chat Application is a Java-based program that enables two-way communication between a server and a client over the network. It uses Java Socket programming to create a TCP connection for sending and receiving messages in real time. Both server and client applications run from the terminal and allow text-based conversation until either party decides to exit. This project demonstrates the implementation of network communication, I/O streams, and multi-threading in Java.

SOFTWARE REQUIREMENTS:-

Programming Language: Java (JDK 8 or higher)

IDE: VS Code, IntelliJ IDEA, or any Java-compatible editor

Libraries: Uses only Java Standard Library APIs (I/O, Serialization)

Operating System: Platform-independent (Windows, Linux, Mac)

Other Requirements: Network connection (localhost or LAN)

DESIGN:-

The application is structured around the following components.

Server Class: Listens for incoming client connections, receives messages from the client, and sends responses back.

ChatClient Class: Connects to the server and facilitates message exchange.

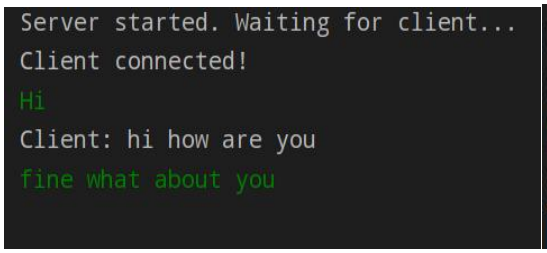
Communication Protocol: TCP sockets are used to ensure reliable delivery of messages.

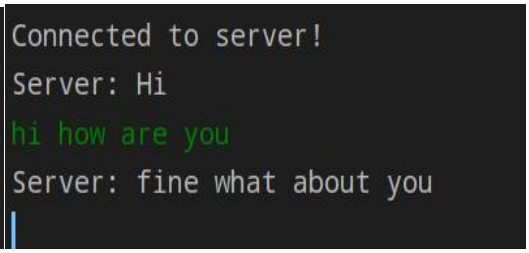
Threading:

1. One thread continuously listens for incoming messages.

2. Another thread waits for user input and sends messages.

INPUT:-





SERVER SIDE **CLIENT SIDE**

IMPLEMENTATION:-

1. Server-side Flow:

1. Create a ServerSocket to listen on a specific port (e.g., 2020).
2. Accept a connection from a client.
3. Start two threads — one for reading and one for writing messages.
4. Continue exchanging messages until exit is received.

2. Client-side Flow:

1. Create a Socket to connect to the server's IP address and port.
2. Start threads for reading and writing messages.
3. Disconnect upon sending or receiving exit

TESTING:-

The Chat Application was tested through the following steps:

1. **Localhost Test:** Server and client run on the same computer using IP 127.0.0.1.

2. **LAN Test:** Client connected to server using the server's local IP in the same network.

3. **Exit Condition Test:** Verified that sending exit from either side ends the connection gracefully.

4. **Multi-message Test:** Checked smooth handling of multiple consecutive messages without delay.

ADVANTAGES:-

1. Real-Time Messaging: Instant text-based communication.

2. Simplicity: Easy to understand and implement.

3. Cross-platform: Runs on any OS with Java installed.

4. No External Libraries Required: Only Java's built-in networking classes are used.

5. Foundation for Advanced Apps: Can be extended for group chats, encryption, or GUI.

CONCLUSION:-

The Chat Application successfully demonstrates real-time two-way communication over TCP using Java. It highlights key concepts such as socket programming, multi-threading, and I/O stream handling. The project forms a good basis for developing more advanced communication systems, like multi-client chat rooms or secure peer-to-peer messengers.

REFERENCES:-

GeeksForGeeks: Java Socket Programming Tutorials
https://www.geeksforgeeks.org/socket-programming-in-java/

JavaTPoint: Java Networking, Socket, and I/O
https://www.javatpoint.com/socket-programming

Oracle Documentation: Java SE Networking API (java.net, java.io)
https://docs.oracle.com/javase/8/docs/api/