**C-Chat - Network Chat Application.**

**Architecture Documentation**

**Author:** Stefan Penchev

**Date:** July 24, 2024

**Introduction**

**Project Purpose:**

C-Chat project aims to develop a simple and efficient network chat console application that allows multiple users to communicate, share files in real-time, meanwhile stores the messages in encrypted format. The primary purpose of this project is to demonstrate skills in network communication and C programming.

**Scope:**

This project focuses on creating a basic chat application using C

programming language and socket programming. It covers the development of both server and client applications, managing the establishment of connections, message broadcasting, and reception of messages. Key features include real-time communication, file sharing, and encrypted message storage.

The project will utilize Docker to containerize the server and client applications, ensuring a consistent and portable environment for development, testing, and deployment. However, the scope does not include advanced features such as graphical user interfaces.

**Technologies Used:**

* **Programming Language:** C
* **Database:** MySQL (8.3.0v)
* **Networking:** TCP/IP Sockets
* **Concurrency:** Pthreads (POSIX Threads)
* **Containerization:** Docker

**Project Motivation:**

The C-Chat project is motivated by the desire to showcase a diverse set of technical skills and competencies through the development of a robust network chat application. This project provides a practical platform to demonstrate expertise in several critical areas of software development:

**Programming Language: C**

* C is a powerful and efficient programming language widely used in systems programming and application development. This project highlights the ability to write efficient, low-level code to manage network communications, file I/O, and memory management, showcasing a deep understanding of C's capabilities and best practices.

**Database: MySQL (8.3.0v)**

* MySQL is a popular relational database management system known for its reliability, scalability, and ease of use. By integrating MySQL for storing user data and chat histories, this project demonstrates proficiency in database design, SQL queries, and the use of database management systems to handle and retrieve data efficiently.

**Networking: TCP/IP Sockets**

* TCP/IP socket programming is essential for enabling network communication between devices. This project illustrates the ability to implement socket programming in C to establish reliable connections, transmit data, and handle multiple client connections concurrently. It demonstrates a thorough understanding of network protocols and communication mechanisms.

**Concurrency: Pthreads (POSIX Threads)**

* Pthreads provide a standardized way to create and manage threads, enabling concurrent execution within a program. This project uses Pthreads to handle multiple client connections simultaneously on the server, ensuring responsive and efficient communication. It showcases the ability to implement multithreading, manage synchronization, and prevent race conditions.

**Containerization: Docker**

* Docker is a powerful tool for creating, deploying, and managing containerized applications. By using Docker to containerize the server and client applications, this project demonstrates knowledge of containerization techniques, ensuring a consistent and portable environment for development, testing, and deployment. It highlights the ability to work with modern DevOps tools to streamline the development workflow.

**Project Overview**

**Project Description:**

The C-Chat project is a network chat console application designed to facilitate real-time communication between multiple users. The application supports text messaging, file sharing, and ensures secure communication by storing messages in an encrypted format.

**Key Features:**

* Real-time text messaging between multiple users
* Groups creating for focused discussions
* Colorized console for distinction of users
* File sharing capibilities
* Encrytped message storage
* Concurrent handaling of multiple client connections
* Containerized deployment

**System Requiredments:**

* **Hardware:** Standard PC with network connectivity
* **­Operating System:** Linux/Unix-based OS recommended
* **Software:**
  + GCC(GNU Compiler Collection) for compiling C code
  + MySQL 8.3.0 version or higher
  + Docker installed for containerization

**Use cases:**

* **Educational Tool**: C-Chat can be used in educational settings to demonstrate network programming, concurrency, and database integration.
* **Project Collaboration**: Teams working on specific projects can create groups to focus discussions and share relevant files securely.

**Software Design**

**Components and Their Responsibilities:**

1. **Server**

* **Socket initialization:** Creates and binds the server socket, listens for incoming connections.
* **Client management:** Accepts new client connections and manages connections.
* **Message Handling:** Receives new messages from clients and broadcasts them to other clients.
* **File Handling:** Manages file sharing between clients.
* **Encryption:** Encrypt messages before storing them.

1. **Client**
   * **Socket Initialization**: Creates a client socket and connects to the server.
   * **User Interaction**: Sends user messages to the server and displays messages received from the server.
   * **File Handling**: Sends and receives files to and from the server.
   * **User Interface**: Provides a command-line interface for interaction.

**Data Flow:**

1. **Connection Establishment**
   * Clients connect to the server using TCP/IP sockets.
   * The server accepts incoming client connections and spawns a new thread for each client.
2. **Message Transmission**
   * Clients send text messages to the server.
   * The server receives messages, processes them, and broadcasts them to all connected clients.
   * Clients receive messages from the server and display them to the user.
3. **File Sharing**
   * Clients can send files to the server.
   * The server receives the file and distributes it to the intended recipients or broadcasts to all clients.
4. **Encryption**
   * Messages are encrypted by the server before storing them in the database.

**Communication Protocol:**

1. **TCP/IP Sockets**
   * Establishes reliable communication channels between clients and the server.
2. **Message Format**
   * Plain text messages, terminated with a newline character for easy parsing.
3. **Concurrency Handling**
   * The server uses Pthreads to manage multiple client connections simultaneously.