**Multi Party Conference Chat**

**(MPCC)**

**Design Document\_MPCC\_v.01**

**Document Control :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | | |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
| 9.8.2022 | 0.1 | Abhishek, Akash, Anjali, Ramani, Vijay | Class diagram, use case diagram, and some theory parts. | | | | Arti Sen | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |

|  |
| --- |
| **Team Members** |

|  |  |
| --- | --- |
| Employee ID: | Name |
| 46238957 | ABHISHEK KUMAR CHOUBEY |
| 46240984 | ANJALI MEHTA |
| 46240549 | AKASH KUMAR MAHABHOI |
| 46240554 | VIJAY KUMAR CHAURASIYA |
| 46240971 | RAMANI UDAGUNDLA |

|  |
| --- |
| **Table of Contents** |

1. **High Level Design**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1.Introduction | | |  |  |
| 1.1 | Purpose | | 6 |  |
| 1.2 | Scope | | 6 |  |
| 1.3 | Definitions | | 6 |  |
| 1.4 | Overview | | 6 |  |
| 2.General Description | | |  |  |
| 2.1 | Product Perspective | | 7 |  |
| 2.2 | Tools used | | 7 |  |
| 2.3 | General Constraints | | 7 |  |
| 2.4 | Assumptions | | 7 |  |
| 2.5 | Special Design aspects | | 7 |  |
| 3.Design Details | | |  |  |
| 3.1 | Main Design Features | | 8 |  |
| 3.2 | Data Flow Diagram | | 8-9 |  |
| 3.3 | Standards | | 10 |  |
| 3.4 | Application Architecture | | 10 |  |
| 3.5 | Files | | 10 |  |
| 3.6 | User Interface | | 10 |  |
| 3.7 | Error Handling | | 11 |  |
| 3.8 | | Help | 11 |  |
| 3.9 | | Performance | 11 |  |
| 3.10 | | Security | 11 |  |
| 3.11 | | Reliability | 11 |  |
| 3.12 | | Maintainability | 12 |  |
| 3.13 | | Portability | 12 |  |
| 3.14 | | Reusability | 12 |  |
| 3.15 | | Application compatibility | 12 |  |
| 3.16 | | Resource utilization | 12 |  |
| 3.17 | | Major Classes | 12 |  |

1. **Low Level Design**

|  |  |
| --- | --- |
| 4. Introduction | |
| 4.1 Purpose 13 | |  |
| 4.2 Document Conventions 13 | |  |
| 4.3 Intended Audience and Reading Suggestions 14 | |  |
| 4.4 References 14 | |  |
|  |
| 5. Detailed System Design  5.1 Design Description 14  5.2 ER Diagram 15  5.3 Class Diagram 16  5.4 Use Case Diagram 17  5.5 Sequence Diagram 18  5.6 Design and Implementation Constraints 19  5.7 User Interface 19 | |  |
|  |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
| 6. Security 19  7.Demo 20,21.23,24 |  |
|  |  |

# 

|  |
| --- |
| **High Level Design** |

**1.Introduction**

**1.1 Purpose**

The purpose of this High Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

**1.2 Scope**

This document provides a comprehensive high level design overview of the Multi Party Conference Chat Application .It highlights the high level flow / use cases in the Chat Application and serves as an input to the low level design documents that would further elaborate on the proposed system design

**1.3 Definitions**

**1.4** **Overview**

This HLD Document is arranged in the following format :

-Section1 : Introduction

A brief explanation about the purpose, aim, scope and design format of the proposed system.

- Section 2 : General Description

This section is all about the general constraints, assumptions and design aspect associated with the proposed system .The product perspective will gives an overall description of the system.

- Section 3 : Design Details

This section documents the detailed design of all modules associated with the development of the proposed system.

1. **General Description**

**2.1 Product Perspective**

The Chat Application provides Clients to connect with the server and exchange messages between them. The system is basically constructed using C++ language. The Server will be able to connect with multiple Clients and host a chat session between them. The User has to register on the Application and then has to log on to it. The system after authentication allows the User to log in and the conversation is initiated which is encrypted. The chat will contain the User’s User ID. The system also has an administrator section that allows the administrator to manage the whole system i.e., after receiving the request from client the sent message could be broadcasted.

**2.2** **Tools used**

1.Ubuntu Terminal.

2.C++ language.

3.GitHub.

4.draw.io

**2.3 General Constraints**

The Multi Party Chat Application should be user-friendly so that people of any age group can use it easily. Data and User privacy should be of utmost priority. The Server should be robust enough to handle multiple Clients.

**2.4 Assumptions**

N.A.

**2.5 Special Design aspects**

1.One of the design aspects is that the Server should be able to host multiple Users.

2.The exchanged messages should be encrypted.

3.The application should always show the active Users’ status.

4.In case the session gets disconnected, the Server will let the Client to rejoin through a new session.

**3 Design Details**

**3.1 Main Design Features**

The main design features include four major parts: the architecture, the user interface design, the files, process relation. In order to make these designs easier to understand, the design has been illustrated in attached diagrams ( Use Case, Data flow diagrams).

**3.2 Data Flow Diagram**

**Zero Level DFD**

**Diagram

Description automatically generated**

**First Level DFD**

**Diagram

Description automatically generated**

**Second Level DFD**

**Diagram

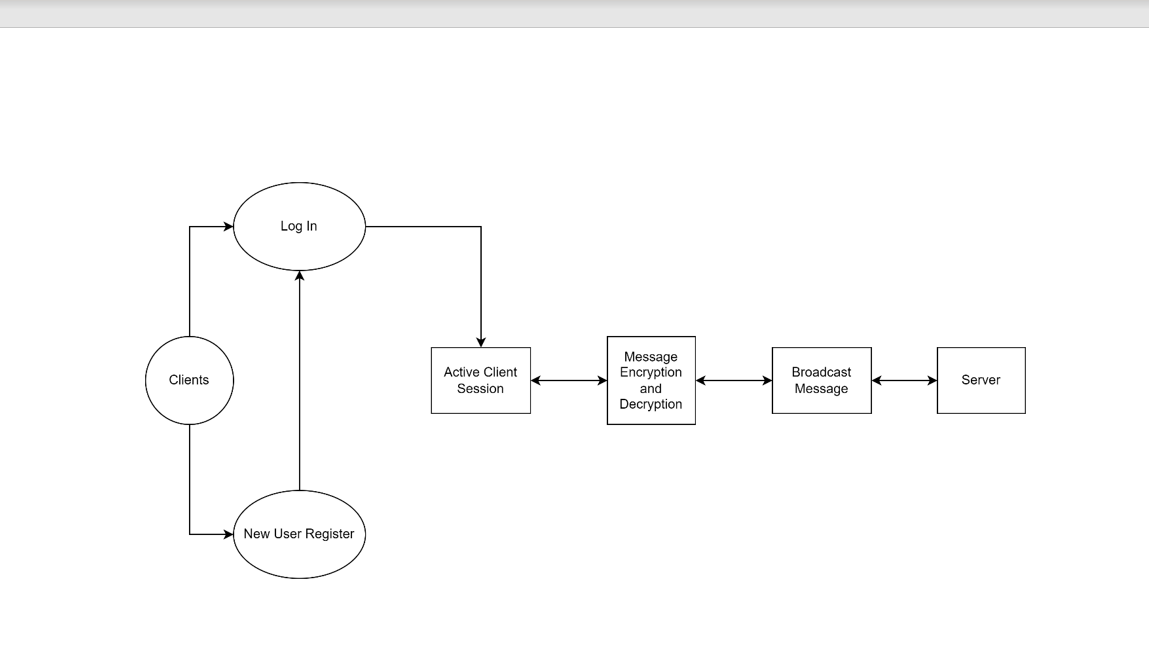
Description automatically generated**

**3.3Standards**

Security – username and password are required for access to the Application.

Quality – by keeping the interface simple and direct, quality should be kept at a maximum.

**3.4 Application Architecture**

****

**3.5 Files**

The MPCC will use number of files for saving data. It will store login data and user data.

**3.6 User Interface**

1. MPCC Application user interface is simple and user friendly to be easily used by every age group.
2. MPCC user interface is consistent. The design is consistent. Increasing consistency increases the familiarity, and hence increases the usability.
3. MPCC Application provides the user with clarity. There is nothing which confuses the user, as it becomes an obstacle for the user in interacting with the Application.

**3.7 Error Handling**

An error will be defined as anything that falls outside the normal and intended usage.

Should errors be encountered, an explanation will be displayed as to what went wrong.

Possible errors

1.User Registration errors- Displayed in case invalid details are provided.

2. Login errors- Displayed incase wrong credentials are provided.

3.Socket error- Displayed in case there is any issue in socket creation.

4.Binding error- Displayed if bind is not successful.

5.Connection error- Displayed if connection is not successful.

6.Listen error- Displayed if listen is not successful.

7.Accept error- Displayed if accept is not successful.

**3.8 Help**

Help will come in the form of all the documentation created prior to coding, which explain the intended uses.

**3.9** **Performance**

Performance is going to be very important for this project. For everything to run smoothly for this project, The Application will work on the user terminal and the performance depends upon the hardware component of the user’s system.

**3.10 Security**

The User’s credentials are not to be shared with anyone. Data encryption is to be done so as to maintain the anonymity of the chat. Password should contain a minimum of 6 characters with at least one special character, one uppercase letter and one number. The application’s back-end files shall never display the User’s password.

**3.11 Reliability**

The application should be able to handle multiple Users and should maintain Concurrency. It should also be robust so as to handle Users during peak hours.

**3.12 Maintainability**

Very little maintenance should be required for this setup. An initial configuration will be the only system required interaction after system is put together. The only other user maintenance would be any changes to settings after setup, and any specified special cases where user settings or history need to be changed. Physical maintenance on the system’s parts may be required, and would result in temporary loss of data or Internet. Upgrades of hardware and software should have little effect on this project, but may result in downtime.

**3.13 Portability**

This system should have the ability that, once it is together, the entire system should be able to be physically moved to any location. Code and program portability should be possible between kernel-recompiled Linux distributions. For everything to work properly, all components should be compiled from source.

**3.14 Reusability**

The code written and the components used should have the ability to be reused with no problems. Should time allow, and detailed instructions are written on how to create this project, everything will be completely reusable to anyone.

**3.15 Application compatibility**

The MPCC is designed as an independent system. As it is not connected to any other components or interfaces, application compatibility is not a concern.

**3.16 Resource utilization**

The MPCC uses very limited resources.

**3.17 Major Classes**

There are a total of 4 major classes: User, Login, Session, Socket.

|  |
| --- |
| **Low Level Design** |

1. **Introduction to MPCC**

The aim of this document is to gather, analyze and give an in-depth insight into the complete Multi Party Conference Chat Application by defining the problem statement in detail. The intended audience includes all stakeholders in the potential system. The detailed low level design of the Multi Party Conference Chat System are provided in this document.

### 4.1 Purpose

The purpose of this document is to describe the low level design flow of a Multi Party Conference Chat Application. The Server will be able to connect with multiple Clients at any given point of time. The User has to register on the system and then has to log on to the Application. The Application after authentication allows the User to log in and the conversation is initiated which is encrypted. The chat will contain the User’s User ID. The system also has an administrator section that allows the administrator to manage the whole system i.e., after receiving the request from client the sent message could be broadcasted. Name of each user will be displayed by mapping IP address. The session should get terminated once the user get disconnected to the session.

### Document Conventions

N.A

### 4.3 Intended Audience and Reading Suggestions

Document is primarily intended for members of **MPCC** team and Mentors which consists of trainees under **Capgemini** Training Program.

**4.4 References**

System Requirement Specification Document.

Project proposal document

High level design of MPCC.

### 5.1 Design Descriptions

The Chat Application provides Clients to connect with the Server and exchange messages between them. The system is basically constructed using C++ language. The Server will be able to connect with multiple Clients.

At the beginning the Client has to register on the system and then has to login on the system as a registered user. The system will check for the details entered by the Client, If found correct it will take the client to the active user area. If the details entered by the Client is found to be wrong, it will prompt a message “incorrect login details”. The client is then logged in and the conversation is initiated between the Client and Server which is encrypted. The chat will display the Client’s User ID and the exchanged chats.

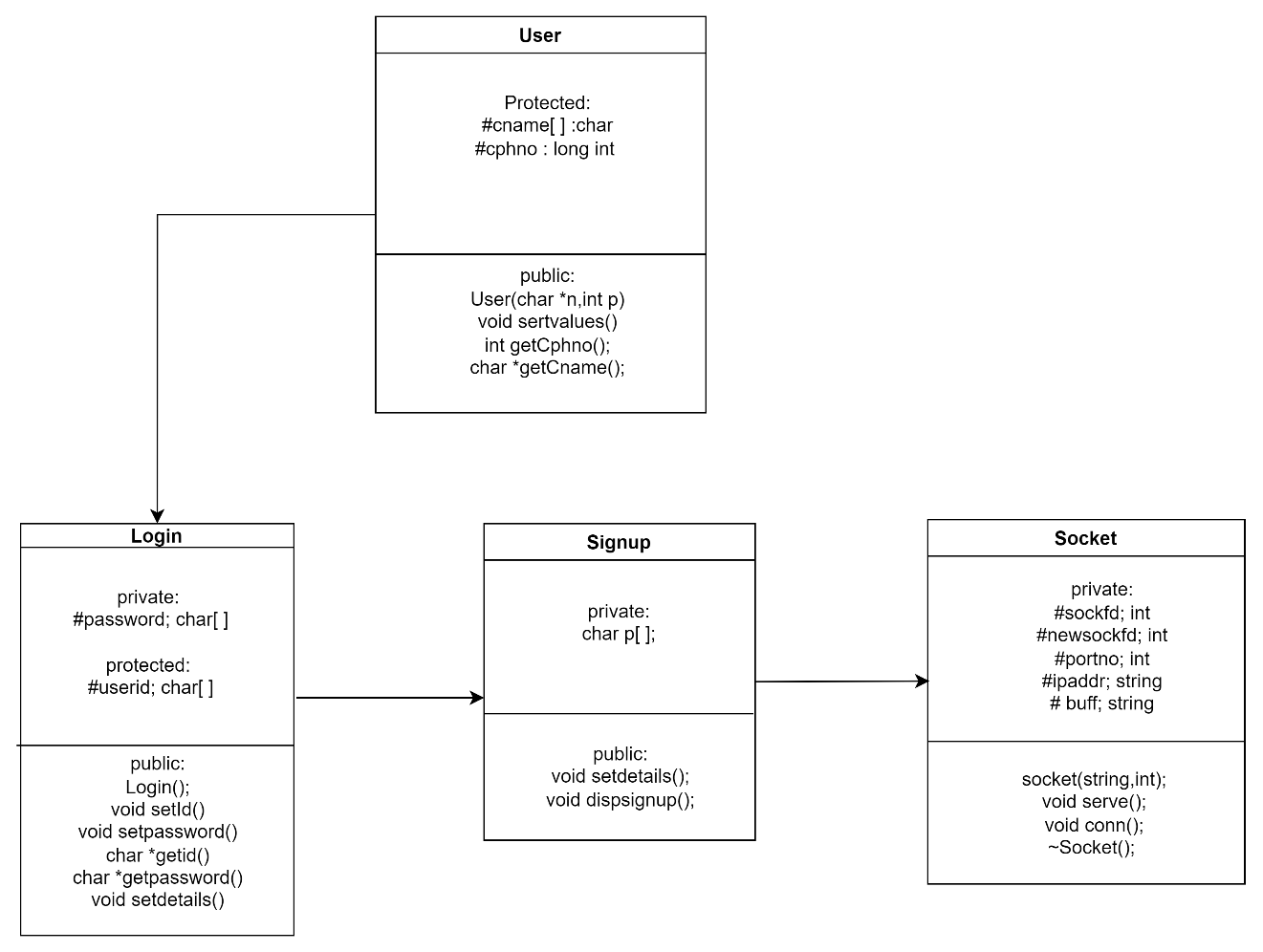
The system also has an administrator section that allows the administrator to manage the whole system i.e., after receiving the request from client the sent message could be broadcasted.

## 5.2 E-R Diagram

Diagram

Description automatically generated

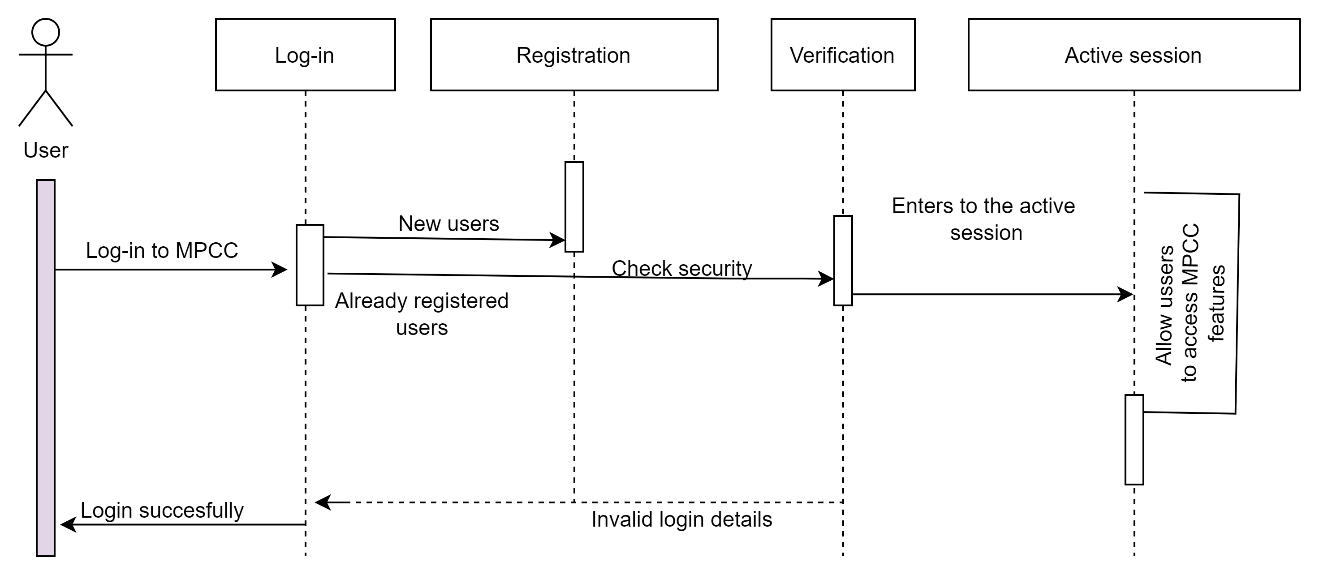
**5.3** **Class Diagram**



**5.4 Use Case Diagram**

## Diagram, schematic Description automatically generated

**5.5 Sequence diagram**



**5.6 Design and Implementation Constraints**

The system is built using C++ language.

### 5.7 User Interface

* MPCC Application user interface is simple and user friendly to be easily used by every age group.
* MPCC user interface is consistent. The design is consistent. Increasing consistency increases the familiarity, and hence increases the usability.
* MPCC Application provides the user with clarity. There is nothing which confuses the user, as it becomes an obstacle for the user in interacting with the Application.

### Security

The customer’s terminal window shall never display a customer’s password as it is encrypted. It shall always be made with special characters representing typed characters. The system’s back-end files shall never display a customer’s password. The customer’s password may be reset but never shown. When the server will receive the user-id, password and the messages they need to decrypt it first to get the actual information.

**7.Demo**

**Welcome page**

**Text

Description automatically generated**

**Registration page**

**Text

Description automatically generated**

**Log-in page**

**Text

Description automatically generated**

**Multiple clients chat**

**Text

Description automatically generated**

**Server page**

**Graphical user interface, text, application

Description automatically generated**