# Introduction

The primary function of this application will be to allow synchronous conferencing between users, who can create chatrooms about certain topics, connect to them and send text to them for others to read. Administration tools will be available to moderate these chat rooms.

# Users

Users who connect are a “guest” account until they sign in. Users will be required to create an account with a username and password to post messages and create chatrooms. This username should consist only of numbers, letters and an underscore. Usernames are limited to a minimum of 6 characters, and a maximum of 20 characters.

# Chat Rooms

Chat rooms are the core of this application, allowing the creation of forums where users can communicate with each other. Any commands executed while in one of these rooms only affects that room. Chatroom names need to be alphanumeric and

## Guest

Guests can see the messages posted in a chatroom, but not post any of their own until they log in.

## Users

Users can connect to chatrooms and send text to them to communicate with the others in the chat room.

## Moderator

Moderators can post to chatrooms like users can, however they can also execute certain commands, outlined in the client commands section.

# Client Commands

These are the commands the client can execute, detailed with the expected response and a list of responses from the server. Responses from the server will consist of a sting that any client can display to its user.

## Command syntax

Client commands will be prefixed by two slashes, and they should be written all in lower-case, with no spaces between the // and the command.

//command

Where there is a parameter, the parameter should be prefixed with an @ symbol and there should not be any spaces between the @ and the previous command, it should be one continuous string.

//command@parameter

For certain commands, parameters can be chained together. Each new perimeter should begin with an @, with the content of the parameter immediately after.

//command@parameter1@parameter2

Certain commands take a string of information. In the message command, any information after the colon is sent to the users inbox.

Unprefixed text will be taken as a chat command and posted to the chat room that the user is currently a part of, if applicable. If the user is not part of a chat room (and the command is not prefixed with a //) then the server will return code “*Command not recognised (incorrect syntax or no parameter specified)*”.

## Non-contextual responses

These are responses that can occur at any time and indicate an error that is not specific to the command sent.

*command\_not\_recognised* (incorrect syntax or no parameter specified) *time\_out*  
*access\_denied* (Occurs when a user tries to execute a command they do not have permission to use)

## Guest

Guests do not have to log on to see public chat rooms, they also have very limited access to the server. All users who connect to the server are guests by default unless they log in.

### //login@exampleuser@password

This command allows the user to log in to a previously defined user account, if the password is correct.

Server acknowledgment: ok  
Server denial: *user\_not\_found* OR *user\_badpass*

### //setup@exampleuser@password

Attempts to set up a user account with the username “exampleuser” and if the username is free it will set up a new account with the specified password.

Server acknowledgment: *login\_ok*Server denial: *preexisting\_user*

### //chatroom@chatroomname

Will move the user to the specified chatroom.  
Server acknowledgment: *ok*Server denial: *no\_chatroom*

## Users

Users will be able to execute a limited number of commands, only those needed by them to use the service. They have access to all the commands that the guests have.

### //msg@user:[message]

Sends the string after the colon to the specified user’s inbox.

Server acknowledgment: *ok*Server denial: *not\_delivered*

### //inbox

Displays all messages currently in the user’s inbox, as well as who they are sent by.

### //createroom@roomname

Creates a chatroom with the specified name and moves the user to it.

Server acknowledgment: *ok*Server denial: *pre-existing\_room*Server denial: *bad\_syntax*

### //logout

Logs the current user out of the system.

Server acknowledgment: *ok*

## Moderator

Access to all the previous commands and access to some additional commands for server-wide administration. In addition, moderators can access whitelisted and blacklisted chatrooms as though they are

### //kick@username.X

Adds the user to a server-wide blacklist and adds their IP to a second list that is also banned. These precautions won't keep a determined troll out, but it will add an extra layer of protection. Admin can specify a number in place of the X to make the ban only last for X number of days. Any user affected by this will be instantly blocked from accessing the server.

### //Revoke@username.X

Adds the user specified to a list which revokes most of their user privileges (essentially making them a guest with a username). Admin can specify a number in place of the X to make the revoke only last for X number of days.

### //blacklist@username

If no parameter is given, then this command returns a formatted list of users currently blacklisted This list would be stored on the server.

Server acknowledgement: *user@user@user@user@user@...*Server denial: *no\_users\_blacklisted*

Adding a parameter to this command adds the username to a list of users who are denied access to the chatroom. The user should not be able to ban themselves in this way, nor should they be able to bad other administrators. Re-entering the command with the same user’s name removes them from the list. This list would be stored on the server

Server acknowledgement: *user\_b\_listed*Server denial: *user\_not\_found*

### //whitelist@username

If no parameter is given, then this command returns a formatted list of users currently whitelisted.

Server acknowledgement: *(information requested)*Server denial: *no\_users\_whitelisted*

Adding a parameter to this commands adds the username to a list of users who can access the chatroom. The first time this command is used the program should automatically add the moderator to the whitelist.

The user should not be able to ban themselves in this way. Re-entering the command with the same user’s name removes them from the list.

Server acknowledgement: *user\_w\_listed*Server denial: *user\_not\_found*

### //closeroom@roomname

Closes the chatroom specified, or the current chatroom if no parameter is given.

Server acknowledgement: *room\_closed*Server denial: *room\_not\_found*