Design Specification Document

Authors:

Group 6:

1.Nishad Aherrao

2.Alexander Windeler

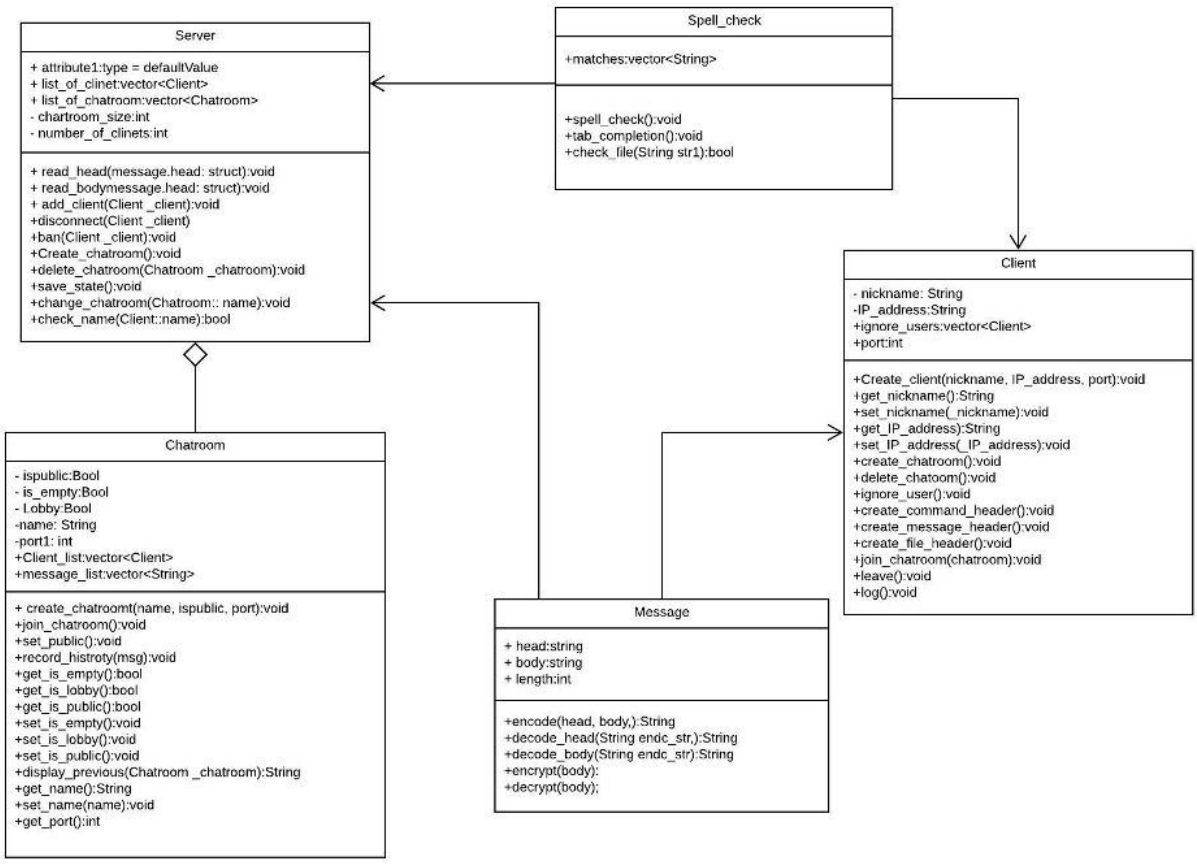
3.Bishesh Shrestha

4.Khom Sitaula

5.Shivang

March 23, 2019.

1.Class Diagram



2.Classes:

Server

The server class is used to handle variables and actions outside of the scope of a single chat room. Only one instance of a server class will exist at a time. The class is used to handle new users connecting to the server by ensuring every client name and by handling connections to clients when they disconnect from the server / log out. The server also handles all of the chat rooms, keeping check on the amount of chat rooms that are open and active, and creating and deleting chat rooms. The class will use the asio library to handle connections to clients and chat rooms.

Attributes:

+list_of_client: vector<Client _clients>: array of all clients currently connected to server

+list_of_chatroom:vector<Chatroom _chatrooms> :array of all chatrooms that exist in the server

-chartroom_size:int: size of list of chatrooms

-number_of_clinets: size of list of clients

Methods:

read_head(message.head) : reads the header of a message to determine what type of message it is

read_body(message.body):reads the body of the message

add_client(client)-adds them to the client_array

disconnect(client)-disconnects remove the clients from the list_of_client

ban(client)-bans the client from accessing the server/chatroom
create_chatroom()- it calls the constructor to create chatroom from chatroom class
delete_chatroom() : it deletes the chatroom from the list_of_chatroom vector
Save_state() :void: saves the current state of the server to a file(hidden)
check_name(Client::name):bool check the availability

Chatroom

Each chat room will be the connection point where multiple clients send chat messages to. The chat room is responsible for receiving these messages and routing them to the other users present. Each chat room keeps track of their own attributes and client connections. All the clients are overseen by the server. The lobby is a special type of chat room that is created upon server creation and cannot be deleted unless the server is taken down. There can be up to ten servers at once. The class will use the asio library to handle connections to the server and clients, and for sending messages.

Attributes:

Bool ispublic -boolean variable, determining if chatroom is public (true) or private (false)
Bool isempty-boolean variable, determining if chatroom is empty (true) or has at least one user in it (false)
Bool lobby-tracks if the chatroom is the lobby, true if it is, false if it is a regular chat room
name:String- it says the name of the chatroom give by the user.
Port1:int it saves thes port number of the chatroom
Client_list:vector<Client>: Stores the vector of clients inside the chatroom
message_listLvector<String>: Stores the messages inside the chatroom

Methods:

create_chatroom(name,ispublic,port): constructor of the chatroom class
+join_chatroom():void: Establishesh the new connection to a new port, the port is associated with the new chatroom which the user wants to join.
+set_public():void: changes ispublic variable
+record_histroty(msg):void : push the messages to the list of messages vector
+get_is_empty():bool :gives the value of isempty variable(private)
+get_is_lobby():bool:gives the value of Lobby variable(private)
+get_is_public():bool: gives the value of ispublic variable(private)
+set_is_empty():void: it changes the value of isempty variable(private)
+set_is_lobby():void : void: it changes the value of lobby variable(private)
+set_is_public():void: it chages the value of ispublic variable(private)

+display_previous(Chatroom _chatroom):String : pops the messages from the list_of_messages vector
+get_name():String : gives the value of name variable(Private)
+set_name(name):void : changes the value of name variable(Private)
+get_port():int: gives the value of port number of the chatroom

Client

There will be multiple instances of client class. Each user will have a client object and the variable name of that object will be the nickname of the user. This class will send a signal to the server and chat room using a message object containing various functions such as create/delete chatroom, ignore user etc…. The client is also where users will view messages in the chat room and send messages and transfer files to other users. Each client will have a unique nickname. The class will use ncurses library to create a GUI for the user and supplies buttons for the user to click for different commands, such as create/delete chatroom, ignore user, etc... , and the asio library to handle port connections and sending message objects to the chat room and server.

Attributes:

- nickname: String : store the unique nickname of the client
-IP_address:String store the ip address of the client
+ignore_users:vector<Client> : the list of ignore users by a client
+port:int store the port number of the client

Methods:

+Create_client(nickname, IP_address, port):void : constructor of the client class

+get_nickname():String : give the nickname of the client(Private)

+set_nickname(_nickname):void changes the nickname of the client(Private)
+get_IP_address):String give the ip address of the client(Private)

+set_IP_address(_IP_address):void changes the ip address of the client(Private)

+create_chatroom():void: Send a signal to the server using the command header to tell the server to create a new chatroom with the name stored in the body (header,body sent to the server by using encode funciton in the message class)
+delete_chatoom():void : Send a signal to the server using the command header to tell the server to delete chatroom with the name stored in the body (header,body sent to the server by using encode function in the message class)

+ignore_user():void : adds a user to the ignore_users vector
+create_command_header():void: creates a message with the command header (to send command)
+create_message_header():void creates a message with the message header (to send message)
+create_file_header():void creates a message with the file header (to send files)
+leave_chatroom():void : establishes a new connection with a new port number base on which chatroom th user want to connect to.

+leave():void : disconnects the client from the chatroom.
+log():void: disconnects the client from all chatrooms and the server.

Message

Message class will function to encode and decode the messages to convert a sendable form of message to readable and vice versa. It will also provide user the functionality of encrypting a message and will handle the decryption of message as well. Messages will also be used to send messages to the server about commands the client wants to do, such as creating new chat rooms, deleting them, or disconnecting. Every type of message (chat room, server command, or file transfer message) will have different headers for them to differentiate between them.

Message
Attributes
+ head:string : store the header value of the message
+ body:string store the body value of the message
+ length:int : store the length of the message

Methods:
+encode(head, body):String : encodes the header and the body into a complete message to send to the server (head + body)
+decode_head(String endc_str, type):String : give the value of just the header from the message
+decode_body(String endc_str, type):String give the value of just the body from the message
+encrypt(body): encrypts the body of the message (obfuscation)
+decrypt(body): decrypts the body of the message (obfuscation)

Spell_Check
This will provide the functionalities of auto reply, spell check and tab completion by checking the input against the file of words provided. Spell check will constantly read the user input and compare the input to the dictionary file. The matches vector keeps track of all the words that match the user input. This vector is used to track spell checking and keeps track of words that can be used for autocompletion. It is associated will client and server class. Spell_check will use the readline library read user inputs to be used for spell checking and tab completion, and the ncurses library mark what words are incorrectly spelt and show menus of words that are available for autocompletion.
Attributes:
+Matches< >: A vector of all words from the text document that match what you are typing in. Filled by calling the read_file() command and pulling out all the substrings that match what is being typed
Methods:
spellcheck(input line)
-constant spell checking the user's input compared to the list of words using read_file and will mark the incorrect spellings red.
tab_completion()
-auto complete the user input, used in conjuction with the read_file method
read_file()
-reads through file->finds the required word/s and saves it in an array and returns true if matches is not empty and returns false if matches is empty.
3. Listing

Class	Client/Server/Both
Client	C
Server	S
Message	B
Chatroom	S
spell_check	B

4.List of Requirements

Lists the classes and methods used for functional requirements

Identifier	Requirement	N/F	Source	C/S/B	Notes	Class/Method
1	Messages delivered in less than 1 second from being sent	N	Document	C		
2	Maximum of 50 users online at one time and an admin.	F	Document	S	Admin doesn't count in the 50 users.	Server class, determined by size of list_of_client vector
3	User ability to create both public and private chat rooms	F	Self Decided	C		Chatroom class, create_chatroom() method
4	SuperChat runs on the linux operating system.	N	Document	B		
5	SuperChat will be implemented using -std=c++11	N	Document	S		

6	User can delete empty chat rooms	F	Document	C		Server class, delete_chatroom() method
7	User provides nickname to be used in chat rooms, can be changed every log in	F	Document	C		Client class, Create_client() method, name attribute
8	SuperChat must have two applications, The “Client” and the “Server”.	N	Document	B		
9	The Client and the Server can run on the same or on different computers.	N	Document	B		
10	User can place others on personal ignore lists to stop seeing messages from them	F	Document	C		Client class, ignore_user() method
11	SuperChat admins can ban users from the chat service	F	Self Decided	S		Server class, ban() method
12	User can use multiple clients at once, as long as they have different nicknames	N	Document	C		
13	The user interface for the Client will use the ncurses.	N	Document	C		
14	The server will provide a user that join a chatroom all of the previous messages.	F	Document	S		Chatroom class, display_previous() method
15	Chat room will record all messages sent until deleted.	F	Document	S		Chatroom class, record_history() method
16	Ten chat rooms can exist at one time, including the lobby	N	Document	S		
17	Messages will be spell checked against a file provided by the instructor.	F	Document	C		Spell_check class, spell_check() method
18	The dictionary will have one word per line, and will not be in any order.	N	Document	C		
19	Messages sent will include the nickname of the person sending and a time stamp	F	Self Decided	S		Client class, Get_nickname() will get the nickname of

						the user and will add it to the front of the text.
20	‘Tab completion’ will be used on the client. Suggest the gnu readline library be used..	F	Document	S		Spell_check class, tab_complete() method
21	Persistent information stored across client invocations will be stored in a file named “~.SuperChat”.	F	Document	S		Server class, Save_state()
22	Files can be transferred between users between their home directories.	F	Document	C		Message class, encode()
23	Files can be transferred between users between their home directories.	F	Document	S		Message class, encode()
24	There is no maximum for users per chatroom as long as there are 50 or less users online at one time.	F	Self Decided	S		Server class, check_name() will validate the entry of the client
25	“Message obfuscation” will be supported. An integer, known to clients, will be used to obscure the meaning of the message for clients that don’t know the number. (technique must be very simple!!)	F	Document	S		Message Class, encrypt(),decrypt().
26	The maximum size for files to be transferred is 50 mb	N	Self Decided	B		
27	There is a default chat room called the ‘lobby’. It can not be deleted.	F	Document	S		Server class, create_chatroom() method
28	The client and server must function without failure for at least 30 minutes.	N	Document	B		
29	A given client can ‘ban’ others from being displayed.	F	Document	C		Client class, ignore() method
30	When SuperChat runs. It will bring user to a login page where user can enter nickname and will be logged in if there aren’t 50 users online already. If there are 50 users online	F	Self decided	S		Server class, check_name()

	then it will pop an error message saying that SuperChat cannot accept any more users at the moment.					
31	When SuperChat runs. It will bring user to a login page where user can enter nickname and will be logged in if there aren't 50 users online already. If there are 50 users online then it will pop an error message saying that SuperChat cannot accept any more users at the moment.	F	Self decided	C		Client class, create_client()
32	There will be a button the create a new chatroom	F	Self decided	C		Server, create_chatroom; Client, delete_chatroom;
33	There will be a button to delete and empty chatroom	F	Self Decided	C		Client, delete_chatroom;
34	There will be a leave button that removes a user from a chatroom and places them back in the lobby	F	Self Decided	C		Client, Leave()
35	There will be a logout button that disconnects the user from the server	F	Self Decided	C		Client, Logout()
36	Replies that are repeated 50 times (common replies) or more will be saved and used to help users 'auto reply'.	F	Documen t	S		spell_check class, auto_reply()