High Level Design

Terminology

Chat Room

A virtual environment in which users can post their messages and read the messages
written by other users.

User

A person with a group id and a humorous nickname who interacts with the system.

Registration

The act of recording user details.

Login

The act of signing into the system by the user.

Message

The text which the user delivers. Message content is limited to 150 characters.

Message Frame

A written communication sent between the users of the system. A wrapper for a message.

Communication model

Requests

Send message request

A send message request is initiated by the user, the request is sent to the server, which assinges the message with a unique ID (GUID) and the server's timestamp.

Get 10 messages request

A get message request is initiated after each "send message request" and potentially can be initiated at any time. This type of request is intended to receive the last 10 messages stored on the server.

Actors

Users

A person connected to the chatroom using a client software, for sending and receiving messages. A user is identified by her group ID and a nickname that is unique to her group.

Milestone 1 Low Level Design

classes

logic Layer -**the logic layer will be incharge of managing all actions that the users can make and handling all the logical structure of the chatroom**

Chat-room

this class will get the data from the presentation layer make it ordered, validate it and sending it to the persistent layer for saving and restoring

Atribbutes:

(-) logger //will be in charge of logging the data to a txt file
(-) List<User> users
(-) List<Message> messages
(-) User currentUser
(-) int count_of_new_message
(-) const string url
(-)const string INVALID_NICKNAME
(-) const string INVALID_GROUPID
(-) const string EMPTY_INPUT
(-) const string INVALID_LOGIN
(-) const string ILLEGAL_LENGTH_MESSAGE

Methods:

(+) Start() - preparing the system to start runing
(+) Registration(int groupId, string nickname)
(+) bool Login(int groupId, string nickname)
(+) List<String> MessageManager(bool ascending, string filter,string sort, string groupId,string nickName) - This returns an updated list of messages that are organized according to the user choice of sorting and filter
(-)static List<String> ConvertToString(List<IMessage> updateList) - convert the Imessage messages to string in order to send them rightfully to the presentation
(-) bool IsValidNickname(int groupId, string nickname)
(-) bool IsValidGroupID(int groupId)
(-) bool CheckIfInputIsEmpty((string str)
(+) Login()
(+)logOut()
(+) RetrieveNMessages(int n) - getting N messeges from the server
(+) List<IMessage> DisplayNMessages(int n)
(+) List<IMessage>DisplayAllMessagesFromCertainUser(string groupId, string nickname)
(+) Send(string messageContent)
(+) List<IMessage> SortByNickname(List<IMessage>updatelist, Boolean ascending)

(+) List<IMessage> SortByIdNicknameTimestamp(List<IMessage> updatelist, Boolean ascending)
(+) List<IMessage> SortTimestamp(List<IMessage> updatelist,Boolean ascending)
(+)List<IMessage> FilterByUser(List<IMessage> list,String groupId, String nickname)
(+) List<IMessage> FilterByGroupId(List<IMessage> list, String groupId)

User

A class of a person who interacts with the system.

Attributes:
(-)string groupId
(-)string nickname

Methods:
(+)Save() -Save user's details in the system files
(+) Send(string messageContent)

Message

A class which describes an object which includes:

Attributes:
(-)Guid Id //is determined by the server
(-)string nickname
(-)DateTime Date //is determined by the server
(-)string messageContent //limited to 150 characters
(-)string groupId

methods:
(+) save() //The method is called from the constructor. Every new Message object will be handled by the persistent layer.
(+)CheckValidity(string content)

Presistent Layer -**the persistent layer will connect between the DataBase and the Logic layer and**

MessageHandler

This class will connect between the DataBase and the Logic layer in all that related to the messages part.

methods:
(+) saveToFile(Message message)
(+) List <Message> restoreMessages()

UserHandler

This class will connect between the DataBase and the Logic layer in all that related to the Users part

methods:
(+) saveToFile(User user)
(+) List< User> restoreUsers()

Presentation Layer - **will be in charge of all the presentation part to the user and will manage the communication between the user and the logic layer. it will contain all the windows that the users can see**

MainWindow

- this class will be the first class that the client can see from here they will be able to move for the other window and enter as users

Attributes:

(-) logger //will be in charge of logging the data to a txt file

(-)ChatRoom chatRoom

methods:

(-)Register() - Open the registration window //display the client the information of the registration process and give the logic layer the details that the client gave the system for handling

(-)Login() - Open the Login Window//display the client the information of the login process and give the logic layer the details that the client gave the system for handling.

(-)exit - Close the program if asked.

Login Window

- this class will ask the client for the login details and send them for the logic layer for validation

Attributes:

(-) logger //will be in charge of logging the data to a txt file

(-)ChatRoom chatRoom

methods:

(-) login - If there are no problems, open the ChatRoom window

(-)BackToMain - Open the main window and close this one.

Chat Room Window

- this class will be the main chat room here the user can see the messages, write new ones, and organize the messages according to his wish.

Attributes:

(-) logger //will be in charge of logging the data to a txt file

(-)ChatRoom chatRoom

(-)string nickName

(-) string groupId

(-) string currChose

(-) string sort

(-)string filter

(-)bool ascending

(-) DispatcherTimer dispatcherTimer

methods:

(-)inisializeFilterandSorter() - inisialize the choose boxeses that will appear on the application with the sorting and filter options

(-) dispatcherTimer_Tick() - this is a timer that responsible for updating the new messages from the server every two seconds

(-)logOut - inform the logic layer that a user asked to log out and move the user to the main manu.

(-) send -sends the chatroom the send request and informs the user in case of an invalid message.

(-)FAS -filter and sort - this function occur when the user press the filter and sort buttons and sends the chat room the new order and filter that the user now wants to use.

(-) RadioButton_checked_sorting options() - Three functions - save the user choice of sorting in the class fields until the user want to start the sort .

(-) filterOptions_SelectionChanged() - this will get the users current filtering choice and display him the place to enter the filter details.