MileStone 3

Low 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 who interacts with the system.

Nickname
A familiar or humorous name the user uses to identify himself.

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.

DB
Data base, a data structure that contains all the user and messages data.

Presentation Layer

Composed of four windows each window is a xaml class, in addition each window has an observable model class witch responsible for the binding between the cs code and the xaml.

MainWindow Class

Functionality

Guides the user when he enters the chat.

Fields

ChatRoom - Instance of ChatRoom class, the pipeline between the Business layer and the communication layer.

MainWindowObserableModel – instance of the class that separate the window's cs code from the xaml.

LoadingWindow- case the user logged in successfully we will display the loading window for defined time using a dispatcher timer.

DispatcherTimer – instance of DispatcherTimer, responsible for displaying the LoadingWindow for only three seconds.

Methods

OpenScreenClick – a method that handles the events done by the user, gets an object and event, case the sender is the register button opens a new RegisterWindow.
Case the sender is the login button, calls the ChatRoom's login method, sets and start the timer.

DispatcherTimer_Tick- creates a LoggedInWindow, opens and closes the loading window.

LoadingWindow

Functionality

Show a random Joke to the User while “loading” the chat. Appears for several seconds to give the user the effect that the chat is complicated and takes some time to load.

LoggedInWindow

Functionality

The main chat window. All the communication graphics are in this window.

Fields

MediaPlayer mediaPlayer – an Ojbect that Provides media playback for for the Chatroom
DispatcherTimer dispatcherTimer- Time counter to indicate the user for how long is the song playing
ChatRoom c- an instance of ChatRoom from the business layer.
LoggedInWindowObservableModel _main – instance of the class that separate the window's cs code from the xaml.

List<LinqAction> actionList – list of the all the LinqActions that we will sort the messages by.

String currentSort – the sort that the messages are currently sorted by.

Methods

LoggedInWindow – the class's constructor, sets and start the dispatcherTimer.

IntialSortFilterProps – insert to the actionList all the Linqactions that require to sort the messages by the users wish.

ShowMessages - a function that retrieve messages from the server, and calls the display function. (The dispatcher timer calls the showMessages function every two seconds).

ApplyClick – a function that operates when the user clicks the apply button of the sorters and filter, calls the method that update the linq actions list and the display method.

EditMessage – takes the data message that need to be edited from the sender, opens a new MyDilalog window for the user to input his new text. Calls the ChatRoom's editMessage methods with the message and the input text.

SendMessage- calls the ChatRoom's SendMessage methods and display the message's list with the new message.

Display – gets an updated list by calling the ChatRoom's function with the LinqAction's list, and update the displayed messages list accordingly.

AddAll – update the observable collection in the observable model class to match the new list.

ResetToDefault – reset all the sorters and filter to the default setting.

PressEnter – A function that responds to a press enter events, calls the send message function.

cbChecked - a function that responds when a check box is checked, calls a ChatRoom's methods to check if the user's input is valid. And disable the matching textbox.

Cb_Unchecked - a function that responds when a check box is unchecked, enable the matching textbox.

ScrollToEnd – scroll the listview scroller down to the last message.

Logout- a function that operates when the user clicks the logout button, call the ChatRoom's logout functions, open a new MainWindow, and close the LoggedInWindow.

RegistrationWindow

Functionality

Responsible for registering new Users into the Chat.

Fields

ChatRoom c- an instance of the chatroom from the business layer responsible to Saving the data of new Users.

Methods

OpenScreenClick – a functions that responds to a click event on the registration button, call the ChatRoom's register function.

G_idTb_LostFocus – checks if the user's input (the nickname and group id) already exist and notify him.

ReturnToMain - opens a new MainWindow and close the RegistrationWindow.

PassBox_PasswordChanged – takes the input password each time a char is typed and checks its validity.

MyDialog

Functionality

A user interface for receiving new text for the edited Messages.

Fields

String TitleText- a string that holds the title of window.

String InputText – a string that holds the user input text.

Boolean Canceled – holds true if the message if the request for editing was canceled.

Methods

BtnCancel_Click – handels the press on the 'cancel' button events, changed the Canceled fields accordingly .

CheckValidty – checks the input validity.

BtnOk_Click - handels the press on the 'ok' button events, changed the Canceled fields accordingly and show a message if the input is invalid.

ObserableModel classes

Each window has a matching observable model class, in order to separate between the threads and bind the components properties between the cs code and the xaml.

Each class has an OnPropertyChange function that -notify the model if the property has changed. And getters and setters for each property.

The difference between each class is its fields which represent the properties that its binds.

Business Layer

Responsible for the logics of the software. contains several classes:

Chatroom Class

Functionality

Manages all the chat activities By delegation to User and Message classes.

Fields

User currentUser – Represents the user who is currently logged into the chat room. Defined 'null' when the chat is initialized.

String URL – The server URL address

List<Message> messageList - List of the messages who saved locally.

MessageHandler messageHandler - an object that handles the message's data Saving/getting.

JokesHandler jokesHandler - an objects that retrieve jokes from the jokes file for making the user happy on a blue day.

String lastDt – holds the date time of the newest messages that currently displayed .

Methods

Register –creates a new User object.

Login – creates a new User Object with the given details and call the User Login method. changes the currentUser field from null to that user.

editMessage – gets a message to be edited and its new body, creates a new FuncMessage and call its Replace method.

SendMassage - calls the SendMassage function in the User class,
retrieve the last messages from the server.

Merge – union two message list , in the condition that the list holds only the newest 200 messages.

Logout - Change the state of the user field to null, and saves the user's and

messages list to files in the cd.

DisplayMessages - A method that displays the all messages that already
been saved to the message list .

FilterSortList – a functions that responsible for sorting and filtering the messages list according to a given list of LinqAction. iterating the list and calls the LinqAction's operation method. Returns the new list.

RetriveTenlastMessages - Retrieve all the newest messages that haven't been dislpayed and merge them into the messages list.

resetList – clear the list from all messages and set the lastDt to null.

updateEditable – update all the edit property of the messages stored in the list

AddGroupFilter – get the group id that the list is to be filtered with and send the request to the persistence layer.

AddNicknameFilter - get the nickname that the list is to be filtered with and send the request to the persistence layer.

LinqAction

LinqAction is an abstract class that declares the abstract method Operate, its purpose is to generalize the sort and filter operations on the messages list.
The Operation methods: gets a list of messages and return a new list sorted, depending on the class.

There are seven classes which extends the LinqAction class all off them implements only the Operation method, we will describe there functionalities:

NicknameSort
sort a list by nickname (according to the alphabetic order) , has a Boolean fields that specifies if the list should be ascending or descending.

TimeSort
sort a list by the time stamp , has a Boolean fields that specifies if the list should be ascending or descending.

G_idSort
sort a list by the group id , has a Boolean fields that specifies if the list should be ascending or descending.

TripleSort
sort a list by the group id nick name and time stamp in that order, has a Boolean fields that specifies if the list should be ascending or descending.

User Class

Functionality

holds the user's attributes and responsible for the user's actions.

Fields

int G_id_int - The user's group ID.
String Nickname - A String that represent the user's nickname.
String Password - A String that holds the user's hased password.
UserHandler userHand - an object that handles the user's data saving/getting.

Methods

Create- a static function that call the constructor if the user properties are valid.

SendMsg – creates a new FuncMessage and calls its Send method.

Equals - Compare two Users by their Group ID and their nicknames.

Login - checks if the the given password is correct by calling the UserHandler method, throw exception if it doesn't.

isExist – return true if the user exist in the system

FuncMessage Class

Functionality

holds the message's attributes .

Fields

GUID GUID - Unique ID that each message gets from the communication layer.

DateTime timeStamp - The time the message has been uploaded to the server. Received from the communication Layer .

User user -The user who sends the message.

String body - The message body.

MessageHandler messageHand - an object that handles the message's data Saving/getting.

Boolean Editable – specified if the message can be edited.

Methods

Create- a static function that call the constructor if the message content is valid.

Send – saves the message in the table by calling the MessageHandler save method.

Replace – replace the messages with the same guid by calling the MessageHandler replace method.

DataMessage Class

A class that holds the same fields as FuncMessage but does not have any methods, this is the class that we will send to the presentation layer.

ExceptionHandler Class

Functionality

handles the exception that is thrown for each case.

Methods

G_idExceptionHandeling – throw exception regarding the group id validity.

ValidG_id - gets a string that represent an group id , and checks if it contains only numbers, returns true if so.

isValidPass – get a given string that represent a password and checks the password validation, return true if valid.

NicknameExceptionHandeling -throw exception regarding the nick name validity.

passwordExceptionHandeling - throw exception regarding the password validity.

CheckValidty – gets a string that represent a message body, return true if valid.

Persistence Layer

Responsible for the communication with the DB stores and gets data from the DB. contains the following classes:

MassegeHandler Class

Functionality

Responsible for handleing the data that refers to Messege objects.

Fields

List<IQueryAction> filters – a list that holds all the currently used filters.

Methods

sqlSave - gets an Message object and insert its properties into the correct table executing a sql command.

Replace- gets a message and update its time and content properties in the sql table.

getMessageList – gets a string that represent the time date of the newest message so far , and read from sql table all the messages that have been sent after this time. Returns an list containing all this messages.

AddGroupFilter –gets a group id and create a GroupFilter accordingly, add it to the filters list.

AddNicknameFilter - gets a nickname and create a NickNameFilter accordingly, add it to the filters list.

clearFilters – delete all the filter that are currently in the list.

UserHendler Class

Functionality

Responsible for handleing the data that refers to User objects.

Methods

SqlUserSaver – gets an User object and insert its properties into the correct table executing a sql command.

getUser – gets a nickname and group id that represent a user , and return a string that holds the user hashed password and id by executing a sql command. The string will be empty if the user does not exist.

IQueryAction

Functionality

An interface that handles adding commends to the Query that send to the SQL server.
Have two classes that implements it:

GroupFilter
Add command that filter by Group Id.
NicknameFilter
Add command that filter by Nickname.

JokesHandler

Functionality

Responsible for reading data from the jokes file.

Fields

String Jokes- the path to the jokes file in the cd,

Methods

readFile – creates a list of string from the file and return it.

Outside the layers

Logger Class

Use log4net Logger to maintain a log which will track activities in our project. Saved the logs in “ChatRoomLogger.txt”.