Lewis Shooter (N0499622) – Samuel Cox (N0516005)

Abstract

Creating a method for server/client requests in order to create a social network. This also required using threads, a database and a GUI to provide the functionality. This document provides how it was designed.

Social Network System Project

The Moo Moo Messenger

Contents

[Introduction To The Project 1](#_Toc417500860)

[The Project 1](#_Toc417500861)

[Server/Client Request System 1](#_Toc417500862)

[Server GUI 2](#_Toc417500863)

[Client GUI 2](#_Toc417500864)

[Database 5](#_Toc417500865)

[Server Code 6](#_Toc417500866)

[ChatServer 6](#_Toc417500867)

[ChatServerExt 9](#_Toc417500868)

[NetworkIO 22](#_Toc417500869)

[serverControl 26](#_Toc417500870)

[Client Code 30](#_Toc417500871)

[Chat 30](#_Toc417500872)

[ChatClient 30](#_Toc417500873)

[Login 30](#_Toc417500874)

[Menu 35](#_Toc417500875)

[Profile 36](#_Toc417500876)

[ProfileEdit 39](#_Toc417500877)

[Register 46](#_Toc417500878)

[ScreenManager 52](#_Toc417500879)

[ServerListener 54](#_Toc417500880)

[Wall 59](#_Toc417500881)

# Introduction to the Project

The assigned project was to create a social network system which could deliver a graphical social network client, with a centralised server that provide a Facebook-like service. The centralised server would allow for multiple clients to connect to it, would keep track of all the clients connected and would keep track of the active/online users.

When a client connects to the server, the server will create a new thread to deal with requests they have. The requests will generally check for information, update information, or create information in the underlying database the server has. The server has a user login table to store usernames, and passwords for each use. A user profile table is used to store user profile information and a user friend table to store the relationships between users.

A user must register an account before they can login to the system. Once they have logged on they can edit and change their profile information. Users can send friend requests to each other and open chats between each other.

# The Project

The main components for the social network system are a request system between the client and server, a way to query and change the database in the server and a GUI for the client. The request system between the client and the server provides the transfer of data between the entire system, and is very important.

## Server/Client Request System

The server creates a serversocket, and listens for any incoming connection. When a connection is accepted it creates a new thread for the client to be handled, and when the client connects to the server it creates a new thread to listen to the server. Both the client thread and the server thread handle the input and output from the other. They listen for a request, then read in an array of details for the request. Depending on the request name, a function is called and the details are passed into the function. The structured English for the request processing is as follows:

Request = fromUser.readLine()

RequestLength = fromUser.readLine()

For i = 0 to RequestLength

RequestDetails[i] = fromUser.readLine()

Switch (request)

Case: “Login”

Login(requestDetails)

This request system allows for any request to be sent with any details related to it. The request is switched in order to call a function to process the details sent for the request. In the case provided it would send a request to login, and the details for the login will consist of the username and password of the user trying to log in. The structured English for the login function is as follows:

Login(loginDetails)

Find user with loginDetails[0]

Check if the users password matches loginDetails[1]

Send login confirmation back to the user

The entire request system for both the server and the client use this system, it is very easy to expand upon because a new request requires 1 new function and a switch case. Inside each request function you can write whatever code you want to make it work. This system will use a bufferedInputStream to readlines from a dataOutputStream. The bufferedInputStream readline takes a whole string so anything text based can be sent. For the client thread and the server listener thread, it will loop through while the connection is still active, taking in a request and switching the request cases. This implementation has been used for the ease of adding new requests to either side of the system.

## Server GUI

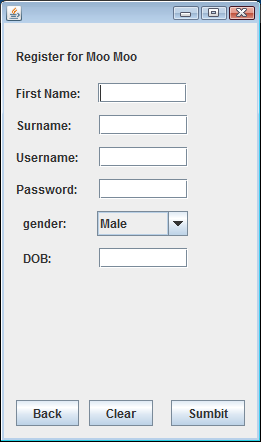


The server GUI was a simple implementation to start the server, stop the server and close the application.

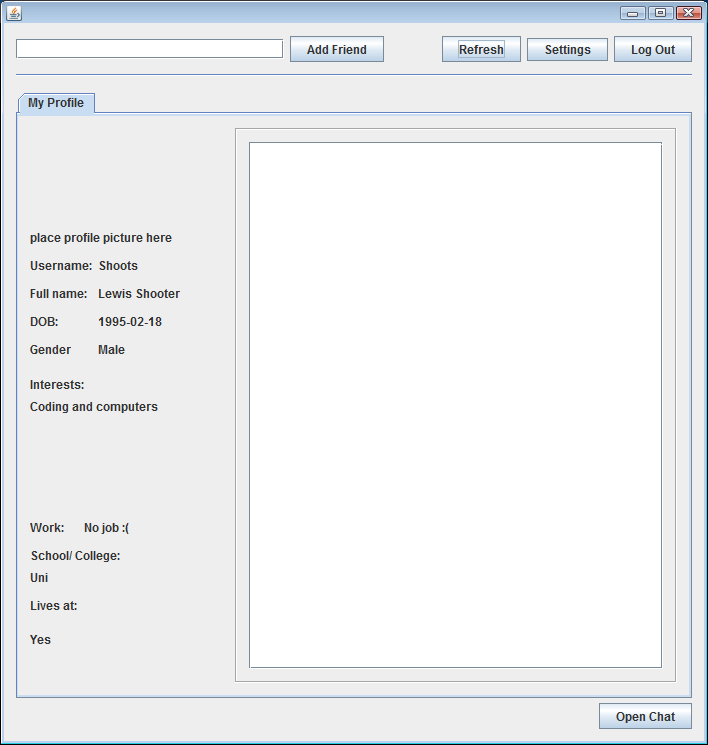
## Client GUI



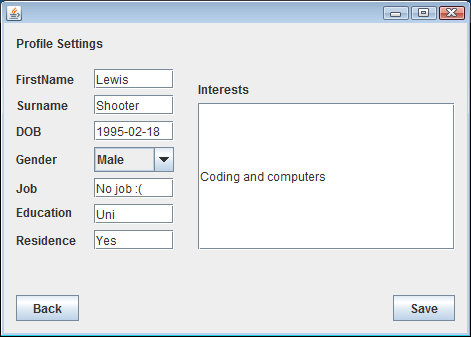
The login screen needed to allow users to send their username, and password to the server, and it also needed a way to let new users register.



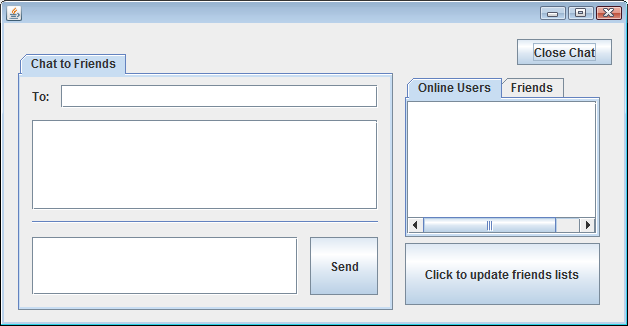
The register screen needed to allow the user to submit their basic information to the server in order to register themselves. The clear button wipes the text fields to empty strings, the back button opens the login menu again, closing the registration form.



The wall needed to show the user information, show information about friend requests, access the other GUI areas such as the profile settings, the chat and the log out.



The profile settings needed to show the user information, and allow the user to edit it. When the user is happy they can save it which will send their updated information to the server.



The chat needed to allow users to send messages to each other. The chat menu also allows for users to see a list of their friends and a list of all online users.

## Database

In the project the database has 3 tables; The user login table, user profile table and user friend table.

* The login table holds information required for users to log in and is accessed when the user tries to log in.
* The profile table holds information about the user profiles and is accessed and changed when the user requests their profile or changes it.
* The friend table holds information about which users are friends and which users have friend requests. When a user tries to send a friend request it is accessed to check any friendship status between the users.

# Server Code

## ChatServer

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package chatserver;

import java.io.\*;

import java.net.\*;

import java.sql.\*;

import java.util.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author N0499622

\*/

public class ChatServer implements Runnable

{

private ServerSocket server;

private TreeMap<String, ChatServerExt> users;

private int clientNum;

public ArrayList<String> activeUsers;

public Connection DBconnection;

public boolean active;

/\*\*

\* @param args the command line arguments

\*/

/\*Name: ChatServer (Constructor)

Description: Opens a new server on the specified port number\*/

public ChatServer(int port)

{

try

{

clientNum = 0;

server = new ServerSocket(port);

users = new TreeMap();

activeUsers = new ArrayList();

connectDatabase("jdbc:derby://localhost:1527/Users", "username", "password");

}

catch (IOException | SQLException ex)

{

}

}

/\*Name: Close

Description: Closes the server\*/

public void Close()

{

try

{

active = false;

server.close();

System.out.println("Closing Server");

}

catch (IOException ex)

{

Logger.getLogger(ChatServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*Name: Listen

Description: This function waits for a connection

then adds a client when a connection comes through\*/

public void run()

{

active = true;

try

{

while(active)

{

System.out.println("Waiting for client." + clientNum);

addClient(server.accept());

}

}

catch (Exception ex)

{

}

}

/\*Name: sendMessageToUser

Description: Relays a message from one user to another

Notes: ID - the reciever

message contains the message sender, reciever and actual message\*/

public void sendMessageToUser(String ID, String[] message)

{

try

{

//Check if the user is online

if (activeUsers.contains(ID))

users.get(ID).netIO.sendRequest("Chat", message);

else

System.out.println("User not active");

}

catch (IOException ex)

{

System.out.println("Chat error!");

Logger.getLogger(ChatServer.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*Name: changeID

Description: Changes the ID on an online user

It deals with making sure the user ID is in the online list and client list\*/

public void changeID(ChatServerExt client, String newID)

{

System.out.println("Client: " + client.ID + " changing to " + newID);

users.remove(client.ID);

System.out.println("Client removed ");

users.put(newID, client);

System.out.println("Client added ");

users.get(newID).ID = newID;

System.out.println("Client set ");

}

/\*Name: activateClient

Description: Adds the user ID to the active user list\*/

public void activateClient(String ID)

{

System.out.println("Activating client: " + ID);

activeUsers.add(ID);

System.out.println("Activated");

}

/\*Name: deactivateClient

Description: Removes the user ID from the active user list\*/

public void deactivateClient(String ID)

{

System.out.println("Deactivating client: " + ID);

activeUsers.remove(ID);

System.out.println("Deactivated");

}

/\*Name: removeClient

Description: Removes the client from the user list\*/

public void removeClient(String ID)

{

users.remove(ID);

}

/\*Name: addClient

Description: Creates and starts a new thread for a client

and adds the client to the list of users\*/

private void addClient(Socket connection)

{

System.out.println("Client Connected.");

System.out.println("Establishing thread.");

ChatServerExt client = new ChatServerExt(this, connection);

Thread thClient = new Thread(client);

String clientNumStr = Integer.toString(clientNum);

users.put(clientNumStr, client);

users.get(Integer.toString(clientNum)).ID = clientNumStr;

clientNum ++;

thClient.start();

}

/\*Name: connectDatabase

Description: Opens a connection with the dataBase

with username and password\*/

private void connectDatabase(String dataBase, String username, String password) throws SQLException

{

/\*This function connects to the database\*/

DBconnection = DriverManager.getConnection(

dataBase,

username,

password);

}

}

## ChatServerExt

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package chatserver;

import java.io.\*;

import java.net.\*;

import java.sql.\*;

import java.util.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author N0499622

\*/

public class ChatServerExt implements Runnable

{

private Socket Sock;

public NetworkIO netIO;

private ChatServer chatServer;

public String ID;

private boolean connected;

/\*Name: ChatServerExt (Constructor)

Description: Sets the socket, chat server and netIO classes for the server extension.\*/

public ChatServerExt(ChatServer server, Socket socket)

{

chatServer = server;

Sock = socket;

ID = "";

try

{

netIO = new NetworkIO(Sock);

}

catch (Exception ex)

{

}

}

/\*Name: run()

Description: Gets a request from the user.

Gets the details related to the request.

Then calls the function that relates to the user request

and passes in the details for the request\*/

public void run()

{

connected = true;

try

{

System.out.println("Socket Established from: " + Sock.getInetAddress());

while (connected)

{

//Get the request and details

String[] request = netIO.readRequest();

System.out.println("Request Read. Copying details ");

String[] requestDetails = Arrays.copyOfRange(request, 1, request.length);

//Call the function that corresponds to the request

System.out.println("Request Read. Switching " + request[0]);

switch (request[0])

{

case "Logout" :

logout();

break;

case "Login" :

System.out.println("Login");

login(requestDetails);

break;

case "Register" :

System.out.println("Register");

register(requestDetails);

break;

case "GetProfile" :

System.out.println("GetProfile");

getProfile(requestDetails);

break;

case "UpProfile" :

System.out.println("UpdateProfile");

updateProfile(requestDetails);

break;

case "FriendRequest" :

System.out.println("RequestFriend");

requestFriend(requestDetails);

break;

case "AcceptFriend" :

System.out.println("AcceptFriend");

acceptFriend(requestDetails);

break;

case "GetFriendsReq" :

System.out.println("GetFriendRequest");

getFriendRequest(requestDetails);

break;

case "GetFriends" :

System.out.println("GetFriendList");

getFriendList(requestDetails);

break;

case "activeUsers" :

System.out.println("activeUsers");

findActiveUsers();

break;

case "Chat" :

System.out.println("Chat");

chatSend(requestDetails);

break;

case "FileRec" :

System.out.println("FileTrans");

netIO.recieveFile("C:/Users/N0499622/Documents/Received.jpg");

break;

case "FileGet" :

System.out.println("FileTrans");

netIO.sendFile("C:/Users/N0499622/Pictures/TOSEND.jpg");

break;

default :

System.out.println("Invalid request");

break;

}

}

Sock.close();

}

catch (Exception ex)

{

chatServer.deactivateClient(ID);

chatServer.removeClient(ID);

System.out.println("Thread error!");

}

}

/\*Name: logout

Description: Removes the client from the active user list

the client list and them sends a logout confirmation

finally it sets the connection to be false in order to disconnect the socket\*/

public void logout()

{

/\*Remove the user from the active user list

the client list and then send a logout confirmation\*/

chatServer.deactivateClient(ID);

chatServer.removeClient(ID);

try

{

netIO.sendRequest("Logout", "true");

}

catch (Exception ex)

{

}

connected = false;

}

/\*Name: login

Description: Looks for the username in the user login table

Then checks the password against the password for that user

It sends a login confirmation to the user.

Notes: loginDetails[0] - Username being checked

loginDetails[1] - Password being checked\*/

public void login(String[] loginDetails)

{

String[] result = new String[2];

result[0] = "false";

try

{

System.out.println("Creating statement");

Statement stmt = chatServer.DBconnection.createStatement();

System.out.println("Starting query");

ResultSet rs = stmt.executeQuery("SELECT Password FROM User\_Logins WHERE Username = '" + loginDetails[0] + "'");

System.out.println("Query completed");

//Check if the user exists

if (rs.next())

{

System.out.println("User found!");

//Check the passwords match

if (rs.getString("Password").equals(loginDetails[1]))

{

System.out.println("Passwords Match!");

result[0] = "true";

result[1] = loginDetails[0];

chatServer.changeID(this, loginDetails[0]);

chatServer.activateClient(ID);

}

else

{

System.out.println("Password wrong.");

}

}

else

{

System.out.println("User not found.");

}

}

catch (SQLException ex)

{

Logger.getLogger(ChatServerExt.class.getName()).log(Level.SEVERE, null, ex);

}

try

{

System.out.println("Login: " + result[0]);

netIO.sendRequest("Login", result);

}

catch (IOException ex)

{

Logger.getLogger(ChatServerExt.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*Name: register

Description: Looks if a user is registered in the user login table

If they don't exist then it creates a new entry in the table & the user profile table

A registration confirmation is send to the user.

Notes: userInfo[0] - Username given

userInfo[1] - Password given

userInfo[2] - First name given

userInfo[3] - Last name given

userInfo[4] - Date of Birth given

userInfo[5] - Gender given\*/

public void register(String[] userInfo)

{

String[] result = new String[3];

result[0] = "false";

try

{

System.out.println("Opening DB");

Statement stmt = chatServer.DBconnection.createStatement();

System.out.println("Executing query");

ResultSet rs = stmt.executeQuery(

"SELECT \* FROM User\_Logins WHERE Username='" + userInfo[0] + "'");

System.out.println("Checking results");

//Check if the user exists

if (!rs.next())

{

System.out.println("User doesn't exist");

stmt.executeUpdate(

"INSERT INTO User\_Logins (Username, Password)\n" +

"VALUES ('" + userInfo[0] + "', '" + userInfo[1] + "')");

stmt.executeUpdate(

"INSERT INTO User\_Profiles (ProfileUsername, First\_Name, Last\_Name, Date\_Of\_Birth, Gender)\n" +

"VALUES ('" + userInfo[0] + "', '" + userInfo[2] + "', '" + userInfo[3] + "', '" + userInfo[4] + "', '" + userInfo[5] + "')");

result[0] = "true";

result[1] = userInfo[0];

result[2] = userInfo[1];

System.out.println("Register Successful.");

}

else

{

System.out.println("Register Failed.");

}

}

catch (SQLException ex)

{

Logger.getLogger(ChatServerExt.class.getName()).log(Level.SEVERE, null, ex);

}

try

{

System.out.println("Register: " + result[0]);

netIO.sendRequest("Register", result);

}

catch (Exception ex)

{

}

}

/\*Name: getProfile

Description: Checks the user exists

then sends the the user info back to the user

Notes: userInfo[0] - Username of profile it returns\*/

public void getProfile(String[] userInfo) //Send image needs to be added

{

String[] profileInfo = new String[1];

profileInfo[0] = "false";

String user = userInfo[0];

System.out.println(user);

//If the username provided is empty then make it this current user

if (user.equals(""))

user = ID;

System.out.println(user);

System.out.println(ID);

try

{

Statement stmt = chatServer.DBconnection.createStatement();

ResultSet rs = stmt.executeQuery(

"SELECT \* FROM User\_Profiles WHERE ProfileUsername='" + user + "'");

//Check if the user exists

if (rs.next())

{

System.out.println("User found!");

profileInfo = new String[10];

profileInfo[0] = "true";

profileInfo[1] = rs.getString("ProfileUsername");

profileInfo[2] = rs.getString("First\_Name");

profileInfo[3] = rs.getString("Last\_Name");

profileInfo[4] = rs.getString("Date\_Of\_Birth");

profileInfo[5] = rs.getString("Gender");

profileInfo[6] = rs.getString("Interests");

profileInfo[7] = rs.getString("Job");

profileInfo[8] = rs.getString("School");

profileInfo[9] = rs.getString("City");

for (String profileInfo1 : profileInfo)

System.out.println(profileInfo1);

}

else

{

System.out.println("User not found.");

}

}

catch (Exception Ex)

{

}

try

{

System.out.println("Sending profile");

netIO.sendRequest("GetProfile", profileInfo);

System.out.println("Sent profile");

}

catch (Exception ex)

{

System.out.println("Try failed");

}

}

/\*Name: updateProfile

Description: Checks the user profile table for the user

then updates the profile table with the new information provided

then sends an update profile confirmation to the user.

Notes: userInfo[0] - Username

userInfo[1] - First name

userInfo[2] - Last name

userInfo[3] - Date of Birth

userInfo[4] - Gender

userInfo[5] - Interests

userInfo[6] - Work/Job

userInfo[7] - Education/School

userInfo[8] - City\*/

public void updateProfile(String[] userInfo)

{

String result = "false";

try

{

Statement stmt = chatServer.DBconnection.createStatement();

ResultSet rs = stmt.executeQuery("SELECT ProfileUsername FROM User\_Profiles WHERE ProfileUsername='" + userInfo[0] + "'");

//Check the user exists

if (rs.next())

{

System.out.println("User found!");

result = "true";

System.out.println("preparing statement");

stmt.executeUpdate("UPDATE User\_Profiles "

+ "SET First\_Name='" + userInfo[1] + "', "

+ "Last\_Name='" + userInfo[2] + "', "

+ "Date\_Of\_Birth='" + userInfo[3] + "', "

+ "Gender='" + userInfo[4] + "', "

+ "Interests='" + userInfo[5] + "', "

+ "Job='" + userInfo[6] + "', "

+ "School='" + userInfo[7] + "', "

+ "City='" + userInfo[8] + "' "

+ "WHERE ProfileUsername='" + userInfo[0] + "'");

System.out.println("User Updated");

}

else

{

System.out.println("User not found.");

}

}

catch (SQLException ex)

{

System.out.println("Profile update failed");

Logger.getLogger(ChatServerExt.class.getName()).log(Level.SEVERE, null, ex);

}

try

{

netIO.sendRequest("UpProfile", result);

}

catch (Exception ex)

{

}

}

/\*Name: requestFriend

Description: Checks if there is a relationship between the users

If there is then it checks if it is a friend request or a friendship

If it is a friend request then it accepts the request

If there isn't then it checks if you already send a friend request

If you haven't then it adds the send request

Then it sends a request friend confirmation to the user

Notes: userInfo[0] - Username of sender

userInfo[1] - Username of reciever\*/

public void requestFriend(String[] userInfo)

{

String[] result = new String[3];

result[0] = "false";

result[1] = userInfo[0];

result[2] = userInfo[1];

try

{

System.out.println("Searching for user");

Statement stmt = chatServer.DBconnection.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM User\_Logins "

+ "WHERE Username='" + userInfo[0] + "'");

//Check I exist

if (rs.next())

{

System.out.println("Searching for relationship");

rs = stmt.executeQuery("SELECT Status FROM User\_Friends "

+ "WHERE FriendUsername='" + userInfo[0] + "' "

+ "And Friend='" + userInfo[1] + "'");

//Check if I have a relationship with you

if (!rs.next())

{

rs = stmt.executeQuery("SELECT Status FROM User\_Friends "

+ "WHERE FriendUsername='" + userInfo[1] + "' "

+ "AND Friend='" + userInfo[0]+ "'");

//Check if I already sent a request

if (!rs.next())

{

//Add request

System.out.println("Sending request");

stmt.executeUpdate("INSERT INTO User\_Friends "

+ "(FriendUsername, Friend, Status) "

+ "VALUES ('" + userInfo[1] + "', '" + userInfo[0] + "', 0)");

result[0] = "Requested";

}

else

result[0] = "AlreadyRequested";

}

else //No relationship

{

//Check if you already sent me a request

if (rs.getInt("Status") == 0)

{

//Accept the request for both users

System.out.println("Accepting request");

stmt.executeUpdate("UPDATE User\_Friends "

+ "SET Status=1 "

+ "WHERE FriendUsername='" + userInfo[0] + "' "

+ "And Friend='" + userInfo[1] + "'");

stmt.executeUpdate("INSERT INTO User\_Friends "

+ "(FriendUsername, Friend, Status) "

+ "VALUES ('" + userInfo[1] + "', '" + userInfo[0] + "', 1)");

result[0] = "Friends";

}

else

result[0] = "AlreadyFriends";

}

}

}

catch (Exception ex)

{

System.out.println("SQL Error");

}

try

{

netIO.sendRequest("FriendRequest", result);

}

catch (Exception ex)

{

}

}

/\*Name: acceptFriend

Description: Looks for a friend request from a user

then accepts it

Notes: userInfo[0] - Username of sender

userInfo[1] - Username of reciever\*/

public void acceptFriend(String[] userInfo)

{

try

{

Statement stmt = chatServer.DBconnection.createStatement();

ResultSet rs = stmt.executeQuery("SELECT Friend FROM User\_Friends "

+ "WHERE FriendUsername='" + userInfo[0] + "' "

+ "And Friend='" + userInfo[1] + "' "

+ "And Status=0");

//If there is a friend request

if (rs.next())

{

//Accept the request for both users

stmt.executeUpdate("UPDATE User\_Friends "

+ "SET Status=1 "

+ "WHERE FriendUsername='" + userInfo[0] + "' "

+ "And Friend='" + userInfo[1] + "'");

stmt.executeUpdate("INSERT INTO User\_Friends "

+ "(ProfileUsername, Friend, Status) "

+ "VALUES ('" + userInfo[1] + "', '" + userInfo[0] + "', 1)");

}

}

catch (Exception ex)

{

}

}

/\*Name: getFriendRequest

Description: Looks for all friend requests

then sends an array of friend requests to the user

Notes: userInfo[0] - Username of whom you want to find requests\*/

public void getFriendRequest(String[] userInfo)

{

ArrayList<String> friends = new ArrayList();

try

{

System.out.println("Checking friend requests");

Statement stmt = chatServer.DBconnection.createStatement();

ResultSet rs = stmt.executeQuery("SELECT Friend FROM User\_Friends "

+ "WHERE FriendUsername='" + userInfo[0] + "' "

+ "And Status=0");

//Get all friend requests

while (rs.next())

{

String request = rs.getString("Friend");

System.out.println("Sending: " + request);

friends.add(request);

}

}

catch (Exception ex)

{

System.out.println("Friend request list caught");

}

try

{

//Send the friend requests to the user

if (friends.size() > 0)

{

System.out.println("Sending true");

friends.add(0, "true");

}

else

{

System.out.println("Sending false");

friends.add("false");

}

netIO.sendRequest("GetFriendsReq", friends.toArray(new String[friends.size()]));

}

catch (Exception ex)

{

System.out.println("Friend request list reply caught");

}

}

/\*Name: getFriendList

Description: Looks for all friends

then sends an array of friends to the user

Notes: userInfo[0] - Username of whom you want to find friends\*/

public void getFriendList(String[] userInfo)

{

ArrayList<String> friends = new ArrayList();

try

{

System.out.println("Checking friends");

Statement stmt = chatServer.DBconnection.createStatement();

ResultSet rs = stmt.executeQuery("SELECT Friend FROM User\_Friends "

+ "WHERE FriendUsername='" + userInfo[0] + "' "

+ "And Status=1");

//Get all friends

while (rs.next())

{

String request = rs.getString("Friend");

System.out.println("Found: " + request);

friends.add(request);

}

}

catch (Exception ex)

{

System.out.println("Friends list caught");

}

try

{

//Send friends to the user

if (friends.size() > 0)

{

System.out.println("Sending true");

friends.add(0, "true");

}

else

{

System.out.println("Sending false");

friends.add("false");

}

netIO.sendRequest("GetFriends", friends.toArray(new String[friends.size()]));

}

catch (Exception ex)

{

System.out.println("Friends list reply caught");

}

}

/\*Name: findActiveUsers

Description: Sends a list of all the active users to the user\*/

public void findActiveUsers()

{

try

{

netIO.sendRequest("activeUsers", chatServer.activeUsers.toArray(new String[chatServer.activeUsers.size()]));

}

catch (Exception ex)

{

}

}

/\*Name: chatSend

Description: Relays a message from one user to another.

Notes: messageDetails[0] - the reciever

messageDetails[1] - the sender

messageDetails[2] - the message\*/

public void chatSend(String[] messageDetails)

{

/\*MessageDetails[0] is who the message is being send to

MessageDetails[1] is the user that sent the message

MessageDetails[2] is the message being send\*/

System.out.println(messageDetails[1] + " says " + messageDetails[2] + " to " + messageDetails[0]);

chatServer.sendMessageToUser(messageDetails[0], messageDetails);

}

}

## NetworkIO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package chatserver;

import java.io.\*;

import java.net.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author N0499622

\*/

public class NetworkIO

{

private Socket Sock;

private InputStream In;

private OutputStream Out;

private DataOutputStream dOut;

private BufferedReader bIn;

/\*Name: NetworkIO (Constructor)

Description: Creates a class for input and output

on a network.

This works on a request system with strings

Notes: This class has been developed to work on both the client

and server side\*/

public NetworkIO(Socket sock) throws Exception

{

Sock = sock;

Out = Sock.getOutputStream();

In = Sock.getInputStream();

dOut = new DataOutputStream(Out);

bIn = new BufferedReader(new InputStreamReader(In));

}

/\*Name: readRequest

Description: Reads a request name

Reads a request size

Reads all the details for the request

Returns an array with the request and the request details\*/

public String[] readRequest() throws Exception

{

//Get the request

String request = bIn.readLine();

System.out.println("Request recived: " + request);

//Get the request size

int requestSize = Integer.parseInt(bIn.readLine());

System.out.println("Request size: " + requestSize);

//Get the details for the request

String[] requestDetails = null;

if (requestSize >= 0)

requestDetails = new String[requestSize+1];

requestDetails[0] = request;

for (int i = 0; i < requestSize; i ++)

{

requestDetails[i+1] = bIn.readLine();

System.out.println("Request detail: " + i+1 + ": " + requestDetails[i+1]);

}

return requestDetails;

}

/\*Name: sendRequest

Description: Sends a request string

a request size string (1)

a single detail string\*/

public void sendRequest(String request, String detail) throws IOException

{

try

{

dOut = new DataOutputStream(Out);

dOut.writeBytes(request + "\n");

int length = 1;

dOut.writeBytes(Integer.toString(length) + "\n");

dOut.writeBytes(detail + "\n");

}

catch (IOException ex)

{

Logger.getLogger(NetworkIO.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*Name: sendRequest

Description: Sends a request string

a request size string (the length of the array)

an array of details strings\*/

public void sendRequest(String request, String[] details) throws IOException

{

try

{

dOut.writeBytes(request + "\n");

int length = details.length;

dOut.writeBytes(Integer.toString(length) + "\n");

for (int i = 0; i < length; i ++)

{

dOut.writeBytes(details[i] + "\n");

}

}

catch (IOException ex)

{

Logger.getLogger(NetworkIO.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*Name: recieveFile

Description: Reads in an array of bytes

Writes the bytes to a file stream to create a file

Notes: This function isn't fully working\*/

public String recieveFile(String fileDest) throws IOException

{

int bytesRead;

int current = 0;

FileOutputStream fos = null;

BufferedOutputStream bos = null;

try

{

//Set up the buffer and output streams

byte[] myByteArray = new byte[6022386];

fos = new FileOutputStream(fileDest);

bos = new BufferedOutputStream(fos);

bytesRead = In.read(myByteArray, 0, myByteArray.length);

int total = bytesRead;

System.out.println("Total bytes: " + bytesRead + " of " + myByteArray.length);

current = bytesRead;

//Read the bytes to the output

System.out.println("Reading bytes");

do

{

bytesRead = In.read(myByteArray, current, (myByteArray.length - current));

if (bytesRead >= 0)

current += bytesRead;

}

while (bytesRead > -1);

//Check if anything was sent

if (total > -1)

{

//Write the file

System.out.println("Writing file");

bos.write(myByteArray, 0, current);

System.out.println("Flushing file");

bos.flush();

System.out.println("File " + fileDest + " downloaded (" + current + " bytes read)");

}

else

{

System.out.println("Failed to read file");

fileDest = "";

}

}

finally

{

//close the outputstreams

if (fos != null)

fos.close();

if (bos != null)

bos.close();

}

return fileDest;

}

/\*Name:

Description: Creates a few file and byte array

Read a byte from the file

Send the file to the client

Notes: This function works with a basic file recieving client

However because the recieve file isn't working then this

isn't used in the main program\*/

public void sendFile(String toSend) throws IOException

{

try

{

//Create the file streams and the array of bytes

File myFile = new File (toSend);

byte[] byteArray = new byte[(int)myFile.length()];

FileInputStream fis = new FileInputStream(myFile);

BufferedInputStream bis = new BufferedInputStream(fis);

//Read in the bytes

bis.read(byteArray, 0, byteArray.length);

System.out.println("Sending " + toSend + "(" + byteArray.length + " bytes)");

//Write the output and send it

Out.write(byteArray, 0, byteArray.length);

Out.flush();

System.out.println("Done.");

}

catch (Exception ex)

{

}

}

}

## serverControl

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package chatserver;

/\*\*

\*

\* @author N0499622

\*/

public class serverControl extends javax.swing.JFrame

{

ChatServer server;

Thread thServer;

/\*\*

\* Creates new form serverControl

\*/

public serverControl()

{

initComponents();

server = new ChatServer(6789);

thServer = new Thread(server);

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

StartServer = new javax.swing.JButton();

StopServer = new javax.swing.JButton();

CloseApplication = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 24)); // NOI18N

jLabel1.setText("Moo Moo Server");

StartServer.setText("Start server");

StartServer.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

StartServerActionPerformed(evt);

}

});

StopServer.setText("Stop server");

StopServer.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

StopServerActionPerformed(evt);

}

});

CloseApplication.setText("Close Application");

CloseApplication.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

CloseApplicationActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(0, 0, Short.MAX\_VALUE)

.addComponent(jLabel1))

.addGroup(layout.createSequentialGroup()

.addComponent(StartServer)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 46, Short.MAX\_VALUE)

.addComponent(StopServer)

.addGap(41, 41, 41)

.addComponent(CloseApplication)))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 54, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(StartServer)

.addComponent(StopServer)

.addComponent(CloseApplication))

.addContainerGap())

);

pack();

}// </editor-fold>

private void CloseApplicationActionPerformed(java.awt.event.ActionEvent evt) {

server.Close();

setVisible(false);

System.exit(0);

}

private void StopServerActionPerformed(java.awt.event.ActionEvent evt) {

server.Close();

}

private void StartServerActionPerformed(java.awt.event.ActionEvent evt) {

if (!server.active)

{

thServer = new Thread(server);

thServer.start();

}

}

/\*\*

\* @param args the command line arguments

\*/

/\*Name: main

Description: Entry point for the program.

This creates the server and runs it\*/

public static void main(String args[])

{

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(serverControl.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(serverControl.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(serverControl.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(serverControl.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable()

{

public void run()

{

new serverControl().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton CloseApplication;

private javax.swing.JButton StartServer;

private javax.swing.JButton StopServer;

private javax.swing.JLabel jLabel1;

// End of variables declaration

}

# Client Code

The client utilises a networkIO class but it is exactly the same as the server version so it isn’t included here.

## Chat

## ChatClient

## Login

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.\*;

import java.net.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author n0516005

\*/

/\*

Name: Login

Description:

Login class constructor Intialises the screen and the GUI componentents.

The lgoin Screen displays two text fields for the username and password for login.

If the user wants to login they must enter their username and password

then press the Login button: LoginMenuActionPerformed.

The username and password are stored in an array then sent using a

a request, this is then sent to the server to "Login".

\*/

public class Login extends Menu

{

/\*\*

\* Creates new form Login

\*/

//Login Constructor initialises the screen using the super class

//and the GUI components are initialised.

public Login(ScreenManager screen)

{

super(screen);

initComponents();

}

/\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

LoginMenu = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

RegisterMenu = new javax.swing.JButton();

jLabel2 = new javax.swing.JLabel();

usernameLogin = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

passwordLogin = new javax.swing.JPasswordField();

Exit = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

LoginMenu.setText("Login");

LoginMenu.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

LoginMenuActionPerformed(evt);

}

});

jLabel1.setFont(new java.awt.Font("Gungsuh", 0, 48)); // NOI18N

jLabel1.setText("Moo Moo Messenger");

RegisterMenu.setText("Register");

RegisterMenu.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

RegisterMenuActionPerformed(evt);

}

});

jLabel2.setText("Username:");

usernameLogin.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

usernameLoginActionPerformed(evt);

}

});

jLabel3.setText("Password:");

Exit.setText("Exit");

Exit.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ExitActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jLabel1)

.addGroup(layout.createSequentialGroup()

.addComponent(Exit)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(RegisterMenu, javax.swing.GroupLayout.PREFERRED\_SIZE, 93, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2)

.addComponent(jLabel3))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(passwordLogin, javax.swing.GroupLayout.PREFERRED\_SIZE, 108, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(usernameLogin, javax.swing.GroupLayout.PREFERRED\_SIZE, 108, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addComponent(LoginMenu, javax.swing.GroupLayout.PREFERRED\_SIZE, 93, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(22, 22, 22)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(usernameLogin, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3)

.addComponent(passwordLogin, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(21, 21, 21)

.addComponent(LoginMenu, javax.swing.GroupLayout.PREFERRED\_SIZE, 49, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 65, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Exit)

.addComponent(RegisterMenu))

.addContainerGap())

);

pack();

}// </editor-fold>

//Login button pressed, get the text from the JTextFields and store them in an array.

//Send request to the server to login, "Login", data (data is an array).

private void LoginMenuActionPerformed(java.awt.event.ActionEvent evt) {

String username, password;

username = usernameLogin.getText();

password = new String(passwordLogin.getPassword());

String[] data = new String[2];

data[0] = username;

data[1] = password;

System.out.println("About to send request");

try

{

screenManager.clientServer.netIO.sendRequest("Login", data);

}

catch (IOException ex)

{

Logger.getLogger(Login.class.getName()).log(Level.SEVERE, null, ex);

}

//set the text of the fields to null.

screenManager.clientServer.username = data[0];

usernameLogin.setText("");

passwordLogin.setText("");

}

//Can register from this menu by clicking the button "RegisterMenu"

//screenMnager sets the screen to the "Register" screen.

private void RegisterMenuActionPerformed(java.awt.event.ActionEvent evt) {

screenManager.setScreen("Register");

}

private void usernameLoginActionPerformed(java.awt.event.ActionEvent evt) {

usernameLogin.setText("");

}

//Shut down client by closing the streams

//see clientServer.java for details

private void ExitActionPerformed(java.awt.event.ActionEvent evt) {

screenManager.clientServer.close();

System.exit(0);

}

// Variables declaration - do not modify

private javax.swing.JButton Exit;

private javax.swing.JButton LoginMenu;

private javax.swing.JButton RegisterMenu;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JPasswordField passwordLogin;

private javax.swing.JTextField usernameLogin;

// End of variables declaration

}

## Menu

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.\*;

import java.net.\*;

/\*\*

\*

\* @author n0516005

\*/

public class Menu extends javax.swing.JFrame {

// Menu class is base class for screen manager.

public ScreenManager screenManager;

public Menu(ScreenManager screen)

{

// ScreenManager instance: screen.

screenManager = screen;

//add other functionality here

}

public void Initialise()

{

}

}

## Profile

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author n0516005

\*/

/\*

Name: Profile

Description:

The profile class is responsible for handling all of the users credentials and

details. The server will send the profile all of the usernames associated data.

The data is read into variables that can be used to populate the users wall.

\*/

public class Profile

{

private String Firstname;

private String Surname;

private String Username;

private String DOB;

private String Gender;

private String Interests;

private String Job;

private String School;

private String City;

ScreenManager screenManager;

public Profile(ScreenManager screen)

{

screenManager = screen;

}

public void Initialise(String[] profile)

{

Username = profile[0];

Firstname = profile[1];

Surname = profile[2];

DOB = profile[3];

Gender = profile[4];

Interests = profile[5];

Job = profile[6];

School = profile[7];

City = profile[8];

//friends list button = profile friend array.

}

/\*

Name: getProfile

Description:

Sends a request to get the profiles credentials.

\*/

public void getProfile()

{

try

{

screenManager.clientServer.netIO.sendRequest("GetProfile", "");

}

catch (IOException ex)

{

Logger.getLogger(Profile.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*

Name: currentUser

Description:

This function is called whenever the curren user logged in needs to use its usernames for requests

\*/

public String currentUser()

{

return Username;

}

/\*

Name: getFullProfile

Description:

When the client needs the full profile to be displayed, the array is returned

to where it is called.

\*/

public String[] getFullProfile()

{

String[] profileData = new String[8];

profileData[0] = Firstname;

profileData[1] = Surname;

profileData[2] = DOB;

profileData[3] = Gender;

profileData[4] = Interests;

profileData[5] = Job;

profileData[6] = School;

profileData[7] = City;

return profileData;

}

public String getSurname()

{

return Surname;

public String getDOB()

{

return DOB;

}

public String getGender()

{

return Gender;

}

//Reset all of the variables to an empty string. Called on Logout.

public void Reset()

{

Username = "";

Firstname = "";

Surname = "";

DOB = "";

Gender = "";

}

}

## ProfileEdit

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author n0516005

\*/

/\*

Name: ProfileEdit

Description:

Constructor intialises the screen using the Super class and the GUI components.

When the Initailiase function is called, the origional prolile data is loaded into

the JTextFields to be edited. This is achieved by gettting the full profile using the function

\*/

public class ProfileEdit extends Menu {

/\*\*

\* Creates new form ProfileEdit

\* @param screen

\*/

public ProfileEdit(ScreenManager screen)

{

super(screen);

initComponents();

}

public void Initialise()

{

System.out.println("Initialising profile edit");

String[] profileInfo = screenManager.profile.getFullProfile();

changeFirstname.setText(profileInfo[0]);

changeSurname.setText(profileInfo[1]);

changeDOB.setText(profileInfo[2]);

changeIntrests.setText(profileInfo[4]);

changeWork.setText(profileInfo[5]);

changeEducat.setText(profileInfo[6]);

changeCity.setText(profileInfo[7]);

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

changeInterests = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

settingsLabel = new javax.swing.JLabel();

FirstNameLabel = new javax.swing.JLabel();

SurnameLabel = new javax.swing.JLabel();

DOBLabel = new javax.swing.JLabel();

GenderLabel = new javax.swing.JLabel();

changeFirstname = new javax.swing.JTextField();

changeSurname = new javax.swing.JTextField();

changeDOB = new javax.swing.JTextField();

changeGenderbox = new javax.swing.JComboBox();

SaveSettings = new javax.swing.JButton();

Back = new javax.swing.JButton();

jLabel2 = new javax.swing.JLabel();

changeWork = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

changeEducat = new javax.swing.JTextField();

changeCity = new javax.swing.JTextField();

jPanel1 = new javax.swing.JPanel();

jLabel5 = new javax.swing.JLabel();

changeIntrests = new javax.swing.JTextField();

changeInterests.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

changeInterestsActionPerformed(evt);

}

});

jLabel1.setText("Interests");

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

settingsLabel.setText("Profile Settings");

FirstNameLabel.setText("FirstName");

SurnameLabel.setText("Surname");

DOBLabel.setText("DOB");

GenderLabel.setText("Gender");

changeGenderbox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Male", "Female", "Other" }));

SaveSettings.setText("Save");

SaveSettings.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SaveSettingsActionPerformed(evt);

}

});

Back.setText("Back");

Back.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

BackActionPerformed(evt);

}

});

jLabel2.setText("Job");

changeWork.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

changeWorkActionPerformed(evt);

}

});

jLabel3.setText("Education");

jLabel4.setText("Residence");

jLabel5.setText("Interests");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(changeIntrests)

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(jLabel5)

.addContainerGap(206, Short.MAX\_VALUE))))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(changeIntrests, javax.swing.GroupLayout.DEFAULT\_SIZE, 147, Short.MAX\_VALUE))

);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addComponent(Back)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 305, Short.MAX\_VALUE)

.addComponent(SaveSettings))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(settingsLabel)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(FirstNameLabel)

.addComponent(DOBLabel)

.addComponent(SurnameLabel)

.addComponent(GenderLabel)

.addComponent(jLabel2)

.addComponent(jLabel3)

.addComponent(jLabel4))

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(changeSurname, javax.swing.GroupLayout.DEFAULT\_SIZE, 80, Short.MAX\_VALUE)

.addComponent(changeDOB, javax.swing.GroupLayout.DEFAULT\_SIZE, 80, Short.MAX\_VALUE)

.addComponent(changeFirstname)

.addComponent(changeGenderbox, 0, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(changeWork)

.addComponent(changeEducat)

.addComponent(changeCity))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGap(0, 0, Short.MAX\_VALUE)))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(settingsLabel)

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(FirstNameLabel)

.addComponent(changeFirstname, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(SurnameLabel)

.addComponent(changeSurname, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(DOBLabel)

.addComponent(changeDOB, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(GenderLabel)

.addComponent(changeGenderbox, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(changeWork, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel3)

.addComponent(changeEducat, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(changeCity, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel4)))

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 45, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(SaveSettings)

.addComponent(Back))

.addContainerGap())

);

pack();

}// </editor-fold>

private void SaveSettingsActionPerformed(java.awt.event.ActionEvent evt) {

String[] data = new String[9];

data[0] = screenManager.profile.currentUser();

data[1] = changeFirstname.getText();

data[2] = changeSurname.getText();

data[3] = changeDOB.getText();

data[4] = changeGenderbox.getSelectedItem().toString();

data[5] = changeIntrests.getText();

data[6] = changeWork.getText();

data[7] = changeEducat.getText();

data[8] = changeCity.getText();

try

{

screenManager.clientServer.netIO.sendRequest("UpProfile", data);

}

catch (IOException ex)

{

Logger.getLogger(ProfileEdit.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void changeInterestsActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void changeWorkActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void BackActionPerformed(java.awt.event.ActionEvent evt) {

screenManager.setScreen("Wall");

}

// Variables declaration - do not modify

private javax.swing.JButton Back;

private javax.swing.JLabel DOBLabel;

private javax.swing.JLabel FirstNameLabel;

private javax.swing.JLabel GenderLabel;

private javax.swing.JButton SaveSettings;

private javax.swing.JLabel SurnameLabel;

private javax.swing.JTextField changeCity;

private javax.swing.JTextField changeDOB;

private javax.swing.JTextField changeEducat;

private javax.swing.JTextField changeFirstname;

private javax.swing.JComboBox changeGenderbox;

private javax.swing.JTextField changeInterests;

private javax.swing.JTextField changeIntrests;

private javax.swing.JTextField changeSurname;

private javax.swing.JTextField changeWork;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JPanel jPanel1;

private javax.swing.JLabel settingsLabel;

// End of variables declaration

}

## Register

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.\*;

import java.net.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author n0516005

\*/

public class Register extends Menu{

/\*\*

\* Creates new form Register

\*

\*/

public Register(ScreenManager screen)

{

super(screen);

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

RegisterTitleLabel = new javax.swing.JLabel();

FirstNameLabel = new javax.swing.JLabel();

SurnameLabel = new javax.swing.JLabel();

usernameLabel = new javax.swing.JLabel();

genderLabel = new javax.swing.JLabel();

ageLabel = new javax.swing.JLabel();

firstname = new javax.swing.JTextField();

surname = new javax.swing.JTextField();

DOB = new javax.swing.JTextField();

username = new javax.swing.JTextField();

GenderBox = new javax.swing.JComboBox();

Submit = new javax.swing.JButton();

PasswordLabel = new javax.swing.JLabel();

password = new javax.swing.JTextField();

Clear = new javax.swing.JButton();

Back = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

RegisterTitleLabel.setText("Register for Moo Moo");

FirstNameLabel.setText("First Name:");

SurnameLabel.setText("Surname:");

usernameLabel.setText("Username:");

genderLabel.setText("gender:");

ageLabel.setText("DOB:");

GenderBox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Male", "Female", "Other" }));

Submit.setText("Sumbit");

Submit.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SubmitActionPerformed(evt);

}

});

PasswordLabel.setText("Password:");

Clear.setText("Clear");

Clear.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ClearActionPerformed(evt);

}

});

Back.setText("Back");

Back.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

BackActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(RegisterTitleLabel, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addComponent(Back)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Clear)

.addGap(18, 18, 18)

.addComponent(Submit))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(FirstNameLabel)

.addGap(18, 18, 18)

.addComponent(firstname, javax.swing.GroupLayout.PREFERRED\_SIZE, 89, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(usernameLabel)

.addComponent(SurnameLabel)

.addComponent(PasswordLabel))

.addGap(21, 21, 21)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(surname, javax.swing.GroupLayout.PREFERRED\_SIZE, 89, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(username, javax.swing.GroupLayout.PREFERRED\_SIZE, 89, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addComponent(password, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 89, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addComponent(ageLabel)

.addGap(46, 46, 46))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addComponent(genderLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(GenderBox, javax.swing.GroupLayout.PREFERRED\_SIZE, 91, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(DOB, javax.swing.GroupLayout.PREFERRED\_SIZE, 89, javax.swing.GroupLayout.PREFERRED\_SIZE)))))

.addGap(0, 57, Short.MAX\_VALUE)))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(RegisterTitleLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 42, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(FirstNameLabel)

.addComponent(firstname, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(SurnameLabel)

.addComponent(surname, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(usernameLabel)

.addComponent(username, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(PasswordLabel)

.addComponent(password, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(genderLabel)

.addComponent(GenderBox, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(ageLabel)

.addComponent(DOB, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 132, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Submit)

.addComponent(Clear)

.addComponent(Back))

.addContainerGap())

);

pack();

}// </editor-fold>

/\*

Name: Register

Description:

The register class is responsible for seding the data taken from the users JTextFields and

then adding them into an array. Once the array has all the inforomation loaded into it's

index, the client sends a request to the server to "Register" the user.

\*/

private void SubmitActionPerformed(java.awt.event.ActionEvent evt) {

String[] data = new String[6];

data[0] = username.getText();

data[1] = password.getText();

data[2] = firstname.getText();

data[3] = surname.getText();

data[4] = DOB.getText();

//data[5] = "male";

data[5] = GenderBox.getSelectedItem().toString();

//data[6] = secretAnswer.getText();

System.out.print("about to send request");

try

{

screenManager.clientServer.netIO.sendRequest("Register", data);

}

catch (IOException ex)

{

Logger.getLogger(Register.class.getName()).log(Level.SEVERE, null, ex);

}

screenManager.clientServer.username = data[0];

System.out.print("sent request");

}

//clear button, sets all components text to empty strings.

private void ClearActionPerformed(java.awt.event.ActionEvent evt) {

DOB.setText("");

username.setText("");

GenderBox.setSelectedIndex(0);

surname.setText("");

password.setText("");

firstname.setText("");

}

//Go back to the loginScreen.

//for infomation on ScreenManager, see ScreenManager.Java.

private void BackActionPerformed(java.awt.event.ActionEvent evt) {

screenManager.setScreen("Login");

}

// Variables declaration - do not modify

private javax.swing.JButton Back;

private javax.swing.JButton Clear;

private javax.swing.JTextField DOB;

private javax.swing.JLabel FirstNameLabel;

private javax.swing.JComboBox GenderBox;

private javax.swing.JLabel PasswordLabel;

private javax.swing.JLabel RegisterTitleLabel;

private javax.swing.JButton Submit;

private javax.swing.JLabel SurnameLabel;

private javax.swing.JLabel ageLabel;

private javax.swing.JTextField firstname;

private javax.swing.JLabel genderLabel;

private javax.swing.JTextField password;

private javax.swing.JTextField surname;

private javax.swing.JTextField username;

private javax.swing.JLabel usernameLabel;

// End of variables declaration

}

## ScreenManager

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.\*;

import java.net.\*;

import java.util.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author n0516005

\*/

public class ScreenManager

{

// tree map of screens that are active.

public TreeMap<String, Menu> screens;

/\*\*

\* @param args the command line arguments

\*/

// Main creats a new instance of screen manager and loads the login screen

public static void main(String[] args)

{

ScreenManager screenie = new ScreenManager();

// set screen is a function defined below that sets the currentMenu to login

screenie.setScreen("Login");

}

Profile profile;

ChatClient clientServer;

Menu[] menuList;

Menu currentMenu;

public Wall wall;

public Chat chatMenu;

/\*

public ScreenManager()

Description:

ScreenManager sets up all the screens and adds them to a map so they can be used whenever the user

wishes to navigate to them. The screenManager is responsible for connecting to the server and setting

the screens of the clients GUI.

clientServer creates a new client (ChatClient) and conntects to the server IP.

set up the profile of the user, the screen (this = Login), a new TreeMap for screens,

set the current menu (currentMenu) to login, and the chat Menu to a new chatMenu (Listen for messages)

set up wall as a new wall:

profile = new Profile(this);

screens = new TreeMap();

currentMenu = new Login(this);

chatMenu = new Chat(this);

wall = new Wall(this);

put ProfileEdit, Login, Register and Wall screens into the map.

screens.put("ProfileEdit", new ProfileEdit(this));

screens.put("Login", currentMenu);

screens.put("Register", new Register(this));

screens.put("Wall", wall);

\*/

public ScreenManager()

{

try

{

clientServer = new ChatClient("MAE109-22.ads.ntu.ac.uk", 6789, this);

}

catch(Exception ex)

{

System.out.println("Client failed");

}

profile = new Profile(this);

screens = new TreeMap();

currentMenu = new Login(this);

chatMenu = new Chat(this);

wall = new Wall(this);

screens.put("ProfileEdit", new ProfileEdit(this));

screens.put("Login", currentMenu);

screens.put("Register", new Register(this));

screens.put("Wall", wall);

}

/\*

public void setScreen(String screenName)

Description:

currentMenu.setVisible(false);

currentMenu = screens.get(screenName);

currentMenu.setVisible(true);

currentMenu.Initialise();

\*/

public void setScreen(String screenName)

{

currentMenu.setVisible(false);

currentMenu = screens.get(screenName);

currentMenu.setVisible(true);

currentMenu.Initialise();

}

public void openChat()

{

chatMenu.setVisible(true);

chatMenu.Initialise();

}

}

## ServerListener

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.net.\*;

import java.io.\*;

import java.util.\*;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author N0499622, N0516005

\*/

/\*

Name: ServerListener

Description:

The server listener listens to the server for a request. When a request is recieved

a Switch statement evaluates the request with a case. For instance; if the server sends

the request "Login". The Login function will be called and the successor of the window

after login will be setup using the ScreenManager.

To see details on the ScreenManager, please see ScreenManager.Java for notes.

\*/

public class ServerListener implements Runnable

{

ChatClient Client;

boolean active;

public ServerListener(ChatClient client)

{

active = true;

Client = client;

}

//Function Run listens for the requests and then reads them when it recieves them.

//the request is determined by reading the first index in the array.

//Switch statment is performed then the appropriate function is called.

public void run()

{

try

{

while (active)

{

String[] request = Client.netIO.readRequest();

String[] requestDetails = Arrays.copyOfRange(request, 1, request.length);

switch (request[0])

{

case "Login" :

Login(requestDetails);

break;

case "GetProfile":

GetProfile(requestDetails);

break;

case "UpProfile":

UpdateProfile(requestDetails);

break;

case "Register":

Register(requestDetails);

break;

case "GetFriends":

GetFriends(requestDetails);

break;

case "activeUsers":

GetActiveUsers(requestDetails);

break;

case "GetFriendsReq":

FriendsRequestList(requestDetails);

break;

case "FriendRequest":

SendFriendRequest(requestDetails);

break;

case "SendMessage":

//call send message function

break;

case "Chat":

chatSend(requestDetails);

break;

case "Logout":

Logout(requestDetails);

break;

case "Disconnect" :

Disconnect();

break;

default :

System.out.println("Invalid request");

break;

}

}

}

catch (Exception ex)

{

}

}

//if login is true, set the screen to the Wall.

public void Login(String[] loginInfo)

{

if (Boolean.valueOf(loginInfo[0]))

{

Client.ScreenManage.setScreen("Wall");

}

//add in login counter for fails

}

/\*

if sending a friend request, switch to which request has been sent.

then update the users wall with the notification.

For example:

case "Friends":

Client.ScreenManage.wall.updateFeed("\n"

+ requestFriend[1] + " is now friends with " + requestFriend[2]);

update the feed using screenManage and print out the message

"your username " + " is now friends with " + " Someones Username";

\*/

public void SendFriendRequest(String[] requestFriend)

{

switch(requestFriend[0])

{

case "Friends":

Client.ScreenManage.wall.updateFeed("\n" + requestFriend[1] + " is now friends with " + requestFriend[2]);

break;

case "Requested":

Client.ScreenManage.wall.updateFeed("\nFriends request sent");

break;

case "AlreadyFriends":

Client.ScreenManage.wall.updateFeed("\nYou are already friends.");

break;

case "AlreadyRequested":

Client.ScreenManage.wall.updateFeed("\nYou have already sent a request.");

break;

default:

Client.ScreenManage.wall.updateFeed("\nThis account has not been found.");

break;

}

}

// print the list of frined requests on the wall feed.

public void FriendsRequestList(String[] userRequests)

{

if(Boolean.valueOf(userRequests[0]))

Client.ScreenManage.wall.PrintList(Arrays.copyOfRange(userRequests, 1, userRequests.length));

}

//Populate a list of all the users information for the profile.

//E.G. Username, Firstname, Interests, Education etc.

public void GetProfile(String[] profileInfo)

{

if(Boolean.valueOf(profileInfo[0]))

Client.ScreenManage.profile.Initialise(Arrays.copyOfRange(profileInfo, 1, profileInfo.length));

}

//if update profile is true (wants to update the profile) call the getProfile() Function.

public void UpdateProfile(String[] profileInfo)

{

if(Boolean.valueOf(profileInfo[0]))

{

Client.ScreenManage.profile.getProfile();

}

}

//If register is true, then the cliennt will log you in.

public void Register(String[] registerInfo)

{

if(Boolean.valueOf(registerInfo[0]))

{

try

{

Client.netIO.sendRequest("Login", Arrays.copyOfRange(registerInfo, 1, registerInfo.length));

//account already exists

//failed text fields

}

catch (IOException ex)

{

Logger.getLogger(ServerListener.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

//when logging out, display the Login menu, set the profile to null and reset the client.

public void Logout(String[] LogoutInfo)

{

if(Boolean.valueOf(LogoutInfo[0]))

{

Client.ScreenManage.setScreen("Login");

Client.ScreenManage.profile.Reset();

Client.Reset();

}

}

//Getting the message that has been sent then printing it on screen.

public void chatSend(String[] messageDetails)

{

Client.ScreenManage.chatMenu.addMessage(messageDetails[1] + ": " + messageDetails[2] + "\n");

}

//Get friends is true, then update the friends array.

public void GetFriends(String[] friendsList)

{

System.out.println("Getting friends");

if(Boolean.valueOf(friendsList[0]))

Client.ScreenManage.chatMenu.updateFriends(Arrays.copyOfRange(friendsList, 1, friendsList.length));

}

//Get active users, call the updateActive function which updates the onlineFriendsList with all users online.

public void GetActiveUsers(String[] onlineFriendsList)

{

System.out.println("Getting active users");

Client.ScreenManage.chatMenu.updateActive(onlineFriendsList);

}

public void Disconnect()

{

active = false;

}

}

## Wall

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package client;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.SwingConstants;

/\*\*

\*

\* @author n0516005

\*/

/\*

Name: Wall

Description:

\*/

public class Wall extends Menu {

ArrayList<String> wallData;

/\*\*

\* Creates new form Wall

\*/

//Constructor set screen using Super class, initialise GUI components

//define and initialise Wall array to empty strings per index.

public Wall(ScreenManager screen)

{

super(screen);

initComponents();

wallData = new ArrayList();

wallData.add("");

wallData.add("");

wallData.add("");

wallData.add("");

wallData.add("");

}

//when initialising the wall, the getProfile Function is called to populate

//the profile infomration with the loggin in users data.

//Set the Text of the users username on the profile and set the editable state of the

//wall to false, this stops the user from writing in the notifcations window.

public void Initialise()

{

//add a wait function

super.Initialise();

screenManager.profile.getProfile();

profileName.setText(screenManager.profile.currentUser());

WallText.setEditable(false);

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

jTextArea1 = new javax.swing.JTextArea();

jCheckBox1 = new javax.swing.JCheckBox();

MyProfileTab = new javax.swing.JTabbedPane();

jPanel1 = new javax.swing.JPanel();

profileName = new javax.swing.JLabel();

ProfilePicture = new javax.swing.JLabel();

ProfileFirstAndSecondName = new javax.swing.JLabel();

ProfileGender = new javax.swing.JLabel();

ProfileDOB = new javax.swing.JLabel();

FeedWall = new javax.swing.JPanel();

jScrollPane2 = new javax.swing.JScrollPane();

WallText = new javax.swing.JTextArea();

InterestsLabel = new javax.swing.JLabel();

ProfileInterests = new javax.swing.JLabel();

workLabel = new javax.swing.JLabel();

ProfileWork = new javax.swing.JLabel();

workLabel1 = new javax.swing.JLabel();

ProfileEducation = new javax.swing.JLabel();

residanceLabel = new javax.swing.JLabel();

ProfileResidance = new javax.swing.JLabel();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

Logout = new javax.swing.JButton();

OpenChat = new javax.swing.JButton();

RefreshButton = new javax.swing.JButton();

Settings = new javax.swing.JButton();

SearchBar = new javax.swing.JTextField();

SearchForFriends = new javax.swing.JButton();

jSeparator1 = new javax.swing.JSeparator();

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jScrollPane1.setViewportView(jTextArea1);

jCheckBox1.setText("jCheckBox1");

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

profileName.setText("Profile user Name");

ProfilePicture.setText("place profile picture here");

ProfileFirstAndSecondName.setText("Profile full name");

ProfileGender.setText("ProfileGender");

ProfileDOB.setText("yyyy-mm-dd");

FeedWall.setBorder(javax.swing.BorderFactory.createEtchedBorder());

WallText.setColumns(20);

WallText.setRows(5);

jScrollPane2.setViewportView(WallText);

javax.swing.GroupLayout FeedWallLayout = new javax.swing.GroupLayout(FeedWall);

FeedWall.setLayout(FeedWallLayout);

FeedWallLayout.setHorizontalGroup(

FeedWallLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(FeedWallLayout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane2, javax.swing.GroupLayout.DEFAULT\_SIZE, 414, Short.MAX\_VALUE)

.addContainerGap())

);

FeedWallLayout.setVerticalGroup(

FeedWallLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(FeedWallLayout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane2)

.addContainerGap())

);

InterestsLabel.setText("Interests:");

ProfileInterests.setText("Interests go here.....\\n");

ProfileInterests.setVerticalAlignment(javax.swing.SwingConstants.TOP);

workLabel.setText("Work:");

ProfileWork.setText("Work goes here");

workLabel1.setText("School/ College:");

ProfileEducation.setText("Education Goes here.....");

residanceLabel.setText("Lives at:");

ProfileResidance.setText("Where do you live?.........");

jLabel1.setText("Username:");

jLabel2.setText("Full name:");

jLabel3.setText("DOB:");

jLabel4.setText("Gender");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(ProfilePicture)

.addComponent(InterestsLabel)

.addComponent(workLabel1)

.addComponent(ProfileEducation, javax.swing.GroupLayout.PREFERRED\_SIZE, 172, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(residanceLabel)

.addComponent(ProfileResidance)

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(workLabel)

.addGap(20, 20, 20)

.addComponent(ProfileWork))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel1)

.addComponent(jLabel2)

.addComponent(jLabel3)

.addComponent(jLabel4))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(ProfileFirstAndSecondName)

.addComponent(profileName, javax.swing.GroupLayout.PREFERRED\_SIZE, 119, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(ProfileDOB)

.addComponent(ProfileGender))))

.addComponent(ProfileInterests, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGap(18, 18, 18)

.addComponent(FeedWall, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(113, 113, 113)

.addComponent(ProfilePicture)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(profileName)

.addComponent(jLabel1))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(ProfileFirstAndSecondName)

.addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(ProfileDOB)

.addComponent(jLabel3))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(ProfileGender)

.addComponent(jLabel4))

.addGap(19, 19, 19)

.addComponent(InterestsLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(ProfileInterests, javax.swing.GroupLayout.PREFERRED\_SIZE, 109, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(ProfileWork)

.addComponent(workLabel))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(workLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(ProfileEducation)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(residanceLabel)

.addGap(18, 18, 18)

.addComponent(ProfileResidance)

.addContainerGap(77, Short.MAX\_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(FeedWall, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addContainerGap())

);

MyProfileTab.addTab("My Profile", jPanel1);

Logout.setText("Log Out");

Logout.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

LogoutActionPerformed(evt);

}

});

OpenChat.setText("Open Chat");

OpenChat.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

OpenChatActionPerformed(evt);

}

});

RefreshButton.setText("Refresh");

RefreshButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

RefreshButtonActionPerformed(evt);

}

});

Settings.setText("Settings");

Settings.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SettingsActionPerformed(evt);

}

});

SearchForFriends.setText("Add Friend");

SearchForFriends.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SearchForFriendsActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(0, 0, Short.MAX\_VALUE)

.addComponent(OpenChat))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(SearchBar)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(SearchForFriends)

.addGap(58, 58, 58)

.addComponent(RefreshButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Settings)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(Logout))

.addComponent(MyProfileTab)

.addComponent(jSeparator1))))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(13, 13, 13)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Logout)

.addComponent(SearchBar, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(SearchForFriends)

.addComponent(Settings, javax.swing.GroupLayout.PREFERRED\_SIZE, 23, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(RefreshButton))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jSeparator1, javax.swing.GroupLayout.PREFERRED\_SIZE, 11, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(MyProfileTab, javax.swing.GroupLayout.PREFERRED\_SIZE, 607, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(5, 5, 5)

.addComponent(OpenChat)

.addContainerGap())

);

pack();

}// </editor-fold>

//open the chat window

private void OpenChatActionPerformed(java.awt.event.ActionEvent evt) {

//screenManager.setScreen("Chat");

screenManager.openChat();

}

//Refresh button is pressed therefore...

//Get the current user from Profile class (See Profile class for details)

//set the profile information fields to the appropriate index

//Request sent to server for getting friends requests.

private void RefreshButtonActionPerformed(java.awt.event.ActionEvent evt) {

String username = screenManager.profile.currentUser();

profileName.setText(username); // change the way chat is called.

String[] profileInfo = screenManager.profile.getFullProfile();

ProfileFirstAndSecondName.setText(profileInfo[0] + " " + profileInfo[1]);

ProfileDOB.setText(profileInfo[2]);

ProfileGender.setText(profileInfo[3]);

ProfileInterests.setText(profileInfo[4]);

ProfileWork.setText(profileInfo[5]);

ProfileEducation.setText(profileInfo[6]);

ProfileResidance.setText(profileInfo[7]);

try

{

//update friend request

screenManager.clientServer.netIO.sendRequest("GetFriendsReq", username);

}

catch (IOException ex)

{

Logger.getLogger(Wall.class.getName()).log(Level.SEVERE, null, ex);

}

}

public void PrintList(String[] List)

{

String wallString = "";

for (String wallData1 : List)

wallString += wallData1 + " Wants to be your friend! Type in their name to accept.\n";

WallText.setText(wallString);

}

private void SettingsActionPerformed(java.awt.event.ActionEvent evt) {

screenManager.setScreen("ProfileEdit");

}

public void updateFeed(String update)

{

wallData.add(update);

wallData.remove(0);

String wallString = "";

for (String WallData1 : wallData)

{

wallString += WallData1 + "\n";

}

WallText.setText(wallString);

}

private void SearchForFriendsActionPerformed(java.awt.event.ActionEvent evt) {

// search for user in all users

String myUser, friendUser;

myUser = screenManager.profile.currentUser();

friendUser = SearchBar.getText();

String[] sendDetails = new String[2];

sendDetails[0] = myUser;

sendDetails[1] = friendUser;

try

{

screenManager.clientServer.netIO.sendRequest("FriendRequest", sendDetails);

}

catch (IOException ex)

{

Logger.getLogger(Wall.class.getName()).log(Level.SEVERE, null, ex);

}

}

//Clear chat and send request to logout.

private void LogoutActionPerformed(java.awt.event.ActionEvent evt) {

try

{

screenManager.chatMenu.Clear();

screenManager.clientServer.netIO.sendRequest("Logout", "");

}

catch (IOException ex)

{

Logger.getLogger(Wall.class.getName()).log(Level.SEVERE, null, ex);

}

}

// Variables declaration - do not modify

private javax.swing.JPanel FeedWall;

private javax.swing.JLabel InterestsLabel;

private javax.swing.JButton Logout;

private javax.swing.JTabbedPane MyProfileTab;

private javax.swing.JButton OpenChat;

private javax.swing.JLabel ProfileDOB;

private javax.swing.JLabel ProfileEducation;

private javax.swing.JLabel ProfileFirstAndSecondName;

private javax.swing.JLabel ProfileGender;

private javax.swing.JLabel ProfileInterests;

private javax.swing.JLabel ProfilePicture;

private javax.swing.JLabel ProfileResidance;

private javax.swing.JLabel ProfileWork;

private javax.swing.JButton RefreshButton;

private javax.swing.JTextField SearchBar;

private javax.swing.JButton SearchForFriends;

private javax.swing.JButton Settings;

private javax.swing.JTextArea WallText;

private javax.swing.JCheckBox jCheckBox1;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JPanel jPanel1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JSeparator jSeparator1;

private javax.swing.JTextArea jTextArea1;

private javax.swing.JLabel profileName;

private javax.swing.JLabel residanceLabel;

private javax.swing.JLabel workLabel;

private javax.swing.JLabel workLabel1;

// End of variables declaration

}