CSC408 Assignment #2

Omar Zeineh (1088546) | Mohammad Fares (1088672) | Hadi R.Z (1088677)

Introduction:

In this assignment we were asked to choose one of the three tasks given, Secure Collaborative Document Review with Queue Management or Math Quiz Challenge with Real-Time Queued Competition or Subscription-Based Reminder System with Queued Access to Events, and we chose the second task which is the math quiz, in this task the objective was to Build a competitive quiz system where clients must enter a virtual queue before participating in a timed math challenge, and the scenario was At scheduled times, quiz sessions open. Clients are placed in a pre-queue until the quiz starts. After shuffling, clients are added to the active challenge queue and are served in FIFO order. And the requirements for this task are:

- Real-time position updates, count-downs, and leaderboard
- Secure session join with encrypted credentials and quiz integrity
- Missed sessions = re-queue; timed sessions prevent queue hogging
- Server must support multiple quiz events with many players

Architecture and Code Description:

The program consists of six classes and one txt file, where there are 2 main classes, and 4 other classes for certain functions and operations, the classes are:

1) Client.java: The client side of the application which attempts to connect to the server using credentials.
2) Server.java: Server side of the application which has the player queue.
3) PlayerInfoThread.java: To create threads to handle user credentials and authentication.
4) QueueThread.java: To create a thread to handle the player queue.
5) GameThread.java: To handle the game logic.
6) Player.java: To create player objects which have the username, password and number of points the user has.
7) Players.txt: Has the player information.

When the server is started the player queue is created and the queue handling thread is created which starts a game immediately if there are at least 5 players or if there are at least 2 players after 60 seconds, then the server waits for sockets to connect to it, and for each socket that connects to it, a thread is created to check if the client's credentials are valid from the player.txt file or not, and if the player is valid checks if the player is already in the player queue. And the game thread handles all the game logic and communication from the server to the clients. And lastly the user has to launch the client using their credentials to connect to the server.

Source Code:

Client:

```
1   package Game;
2
3   import java.net.*;
4   import java.io.*;
5
6   public class Client {
7
8       public static void main(String args[]) {
9           // username args[0]
10          // password args[1]
11          Socket s = null;
12          try {
13              //credintials
14              String message = args[0] + "/" + args[1];
15              s = new Socket("localhost", 55550);
16              DataInputStream in = new DataInputStream(s.getInputStream());
17              DataOutputStream out = new DataOutputStream(s.getOutputStream());
18              //send credintials
19              out.writeUTF(message);
20              String d = in.readUTF();
21              System.out.println(d);
22              boolean done = false;
23              //check if it was able to login
24              if (d.equals("Login successful")) {
25                  //create bufferedreader to not block the program when we want to input
26                  BufferedReader userInput = new BufferedReader(new InputStreamReader(System.in));
27                  while (!done) {
28                      String serverMsg = in.readUTF();
29                      System.out.println(serverMsg);
30                      //check if game started
31                      if (serverMsg.equals("game is starting")) {
32                          while (true) {
33                              String msg = in.readUTF();
34                              System.out.println(msg);
35                              //check if question is addressed to the client
36                              if (msg.equals("Question for " + args[0])) {
```

```
                    String question = in.readUTF();
                    System.out.println("Question: " + question);

                    System.out.print("Your answer: ");
                    //take answer from user
                    String answer = userInput.readLine();

                    out.writeUTF(answer);

                    String result = in.readUTF();
                    System.out.println(result);
                }
                //check if the game is over
                if(msg.equals("Thank you for playing!")) {
                    done = true;
                    break;
                }
            }
        }
    }

    } catch (UnknownHostException e) {
        System.out.println("Error Socket:" + e.getMessage());
    } catch (EOFException e) {
        System.out.println("Error EOF:" + e.getMessage());
    } catch (IOException e) {
        System.out.println("Error readline:" + e.getMessage());
        //} catch (InterruptedException e) {
        //  e.printStackTrace();
    } finally {
        if (s != null) {
            try {
                s.close();
            } catch (IOException e) {
                System.out.println("Error close:" + e.getMessage());
            }
        }
    }
}
}
```

Server:

```
1   package Game;
2
3   import java.util.LinkedList;
4   import java.net.*;
5   import java.io.*;
6
7   public class Server {
8
9       public static void main(String[] args) {
10          //create the user queue using linked list
            LinkedList<player> userQueue = new LinkedList<player>();
            ServerSocket listenSocket = null;
13          try {
14              //create server socket
15              listenSocket = new ServerSocket(55550);
16              //create queue handling thread
                QueueThread QT = new QueueThread(userQueue);
18              while (true) {
19                  //wait for clients to connect
20                  Socket clientSocket = listenSocket.accept();
21                  //handle each connection using a thread to validate the credintials
                    new PlayerInfoThread(clientSocket, userQueue);
23              }
24          } catch (IOException e) {
25              System.out.println("Error Listen socket:" + e.getMessage());
26          }
27      }
28  }
29
```

PlayerInfoThread:

```java
package Game;

import java.net.*;
import java.io.*;
import java.io.File;  // Import the File class
import java.io.FileNotFoundException;  // Import this class to handle errors
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Scanner;

public class PlayerInfoThread extends Thread {

    Socket clientSocket;
    DataInputStream in;
    DataOutputStream out;
    LinkedList<player> userQueue;

    public PlayerInfoThread(Socket clientSocket, LinkedList<player> userQueue) {
        try {
            this.clientSocket = clientSocket;
            this.userQueue = userQueue;
            in = new DataInputStream(clientSocket.getInputStream());
            out = new DataOutputStream(clientSocket.getOutputStream());
            this.start();
        } catch (IOException e) {
            System.out.println("Error Connection:" + e.getMessage());
        }
    }

    @Override
    public void run() {
        try {
            //get the user data from the message sent by the user
            String[] data = in.readUTF().split("/");
            String username = data[0];
            String password = data[1];
```

```
37              boolean userFound = false;
38              boolean userOn = false;
39
40              //synchronize the file reading operation so only one thread can access it at time
41              synchronized (this) {
42                  File file = new File("C:\\Users\\omarz\\OneDrive\\Documents\\NetBeansProjects\\DistHW2\\src\\Game\\Players.txt");
                    Scanner input = new Scanner(file);
44                  while (input.hasNext()) {
                        String[] in = input.nextLine().split("/");
46                      String user = in[0];
47                      String pw = in[1];
48                      //check if username exists and the password is correct
49                      if (user.equals(username) && pw.equals(password)) {
50                          userFound = true;
51                          //check if user is not in player queue
52                          for (int i = 0; i < userQueue.size(); i++) {
53                              if (userQueue.get(i).getUsername().equals(user)) {
54                                  userOn = true;
55                              }
56                          }
57                          break;
58                      }
59                  }
60                  input.close();
61              }
62
63              //if user is found and not in the player queue tell the client that the login was successful
64              if (userFound & !userOn) {
65                  out.writeUTF("Login successful");
66                  //add player to player queue (multiple threads will try to access the userQueue object, so synchronization is needed)
                    synchronized (userQueue) {
68                      userQueue.addLast(new player(username, password, clientSocket));
69                      System.out.println(userQueue);
70                  }
71              } else if (userOn) {
72                  out.writeUTF("Login failed. User already in queue.");
```

```
73              } else {
74                  out.writeUTF("Login failed. User not found.");
75              }
76
77              //} catch (InterruptedException e) {
78              //System.out.println(e.getMessage());
79          } catch (EOFException e) {
80              System.out.println("Error EOF:" + e.getMessage());
81          } catch (IOException e) {
82              System.out.println("Error readline:" + e.getMessage());
83          } finally {
84          }
85      }
86  }
```

QueueThread:

```java
package Game;

import java.io.DataOutputStream;
import java.io.IOException;
import java.util.LinkedList;
import java.util.logging.Level;
import java.util.logging.Logger;

public class QueueThread extends Thread {

    long timer;
    LinkedList<player> userQueue;

    public QueueThread(LinkedList<player> userQueue) {
        this.timer = System.currentTimeMillis();
        this.userQueue = userQueue;
        this.start();
    }

    @Override
    public void run() {
        try {
            while (true) {
                //count how many seconds have passed since the timer was last reset
                int secs = (int) ((System.currentTimeMillis() - this.timer) / 1000);
                //other threads will try to access the userQueue object, so synchronize on it
                synchronized (userQueue) {
                    Thread.sleep(100);
                    //send how many seconds have passed to all clients in the queue
                    for (int i = 0; i < this.userQueue.size(); i++) {
                        DataOutputStream out = new DataOutputStream(userQueue.get(i).getCurrentSocket().getOutputStream());
                        out.writeUTF(String.valueOf(secs));
                    }
                    //check if there are atleast five players inside the queue, if so start a game immediatly and reset the timer
                    if (this.userQueue.size() >= 5) {
                        LinkedList<player> gameQueue = new LinkedList<player>();
                        for (int i = 0; i < 5; i++) {
                            //create a game queue with the first five players in the queue
                            gameQueue.addLast(userQueue.removeFirst());
                        }
                        //create a game thread using the first five players in the queue
                        new GameThread(gameQueue);
                        //reset the timer
                        this.timer = System.currentTimeMillis();
                        //if there are atleast two players and 60 seconds passed without a game starting, start a game, with as many pl
                    } else if (secs == 60 & 2 <= this.userQueue.size()) {
                        LinkedList<player> gameQueue = new LinkedList<player>();
                        int size = this.userQueue.size();
                        for (int i = 0; i < size; i++) {
                            gameQueue.addLast(userQueue.removeFirst());
                        }
                        new GameThread(gameQueue);
                        this.timer = System.currentTimeMillis();
                      // if 60 seconds passed without a game starting reset the timer
                    } else if (secs == 60 & 2 > this.userQueue.size()) {
                        this.timer = System.currentTimeMillis();
                    }
                }
            }
        } catch (IOException e) {
            System.out.println("Error readline:" + e.getMessage());

        } catch (InterruptedException ex) {
            Logger.getLogger(QueueThread.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

GameThread:

```
1   package Game;
2
3   import java.io.DataInputStream;
4   import java.io.DataOutputStream;
5   import java.io.IOException;
6   import java.net.SocketTimeoutException;
7   import java.util.LinkedList;
8   import java.util.logging.Level;
9   import java.util.logging.Logger;
10
11  public class GameThread extends Thread {
12
13      LinkedList<player> gameQueue;
14
15      public GameThread(LinkedList<player> gameQueue) {
        this.gameQueue = new LinkedList<player>();
17      //shuffle the players (non random shuffle)
18      synchronized (gameQueue) {
19          LinkedList<player> reordered = new LinkedList<>();
20          for (int i = 0; i < gameQueue.size(); i++) {
21              player p = gameQueue.get(i);
22              if (i % 2 == 0) {
23                  reordered.addLast(p);
24              } else {
25                  reordered.addFirst(p);
26              }
27          }
28          gameQueue.clear();
29          this.gameQueue.addAll(reordered);
30      }
31
        this.start();
33  }
34
35      @Override
```

```
public void run() {
    try {
        //tell clients that the game is starting
        sendToAll("game is starting");

        //play three rounds
        for (int round = 0; round < 3; round++) {
            //ask a player a question each round
            for (player currentPlayer : gameQueue) {
                //announce the to the clients the player that needs to respond
                sendToAll("Question for " + currentPlayer.getUsername());

                //generate a random question
                int a = (int)(Math.random()*10+1);
                int b = (int)(Math.random()*10+1);
                int correctAnswer = a+b;

                String question = "What is " + a + " + " + b + " ?";
                DataOutputStream out = new DataOutputStream(currentPlayer.getCurrentSocket().getOutputStream());
                DataInputStream in = new DataInputStream(currentPlayer.getCurrentSocket().getInputStream());

                //send the question
                sendToAll(question);
                currentPlayer.getCurrentSocket().setSoTimeout(10000); // 10 seconds timeout (player turn)

                try {
                    //take the players answer
                    String answerStr = in.readUTF();
                    int userAnswer = Integer.parseInt(answerStr);

                    //check if the answer is correct, if it is correct add a point, if not announce the correct answer
                    if (userAnswer == correctAnswer) {
                        currentPlayer.addPoints(1);
                        sendToAll("Correct, good job");
                    } else {
                        sendToAll("Wrong, The correct answer was: " + correctAnswer);
                    }
                    //send the stat board
                    sendToAll(gameQueue.toString());
                } catch (SocketTimeoutException e) {
                    out.writeUTF("You ran out of time, be faster next time.");
                } catch (NumberFormatException e) {
                    out.writeUTF("Invalid input. Numbers only.");
                }
            }
        }

        sendToAll("Game over. Final Scores:");
        //send the scores of each player to all players
        for (player p : gameQueue) {
            sendToAll(p.getUsername() + ": " + p.getPoints());
        }
        //announce the end of the game
        sendToAll("Thank you for playing!");

    } catch (IOException ex) {
        Logger.getLogger(GameThread.class.getName()).log(Level.SEVERE, null, ex);
    }
}

    public void sendToAll(String s) throws IOException {
        for (int i = 0; i < this.gameQueue.size(); i++) {
            DataOutputStream out = new DataOutputStream(gameQueue.get(i).getCurrentSocket().getOutputStream());
            out.writeUTF(s);
        }
    }
```

Player:

```java
package Game;

import java.net.Socket;


public class player {
    private String username;
    private String password;
    private Socket CurrentSocket;
    private int points;

    public player(String username, String password, Socket CurrentSocket) {
        this.username = username;
        this.password = password;
        this.CurrentSocket = CurrentSocket;
        this.points = 0;
    }

    public String getUsername() {
        return username;
    }

    public String getPassword() {
        return password;
    }

    public Socket getCurrentSocket() {
        return CurrentSocket;
    }

    public int getPoints() {
        return points;
    }

    public void setUsername(String username) {
        this.username = username;
```

```
37          }
38
39          public void setPassword(String password) {
40              this.password = password;
41          }
42
43          public void setCurrentSocket(Socket CurrentSocket) {
44              this.CurrentSocket = CurrentSocket;
45          }
46
47          public void setPoints(int points) {
48              this.points = points;
49          }
50
51          public void addPoints(int points) {
52              this.points = this.points + points;
53          }
54
55          @Override
            public String toString() {
57              return username + ": " + points;
58          }
59
60
61      }
```

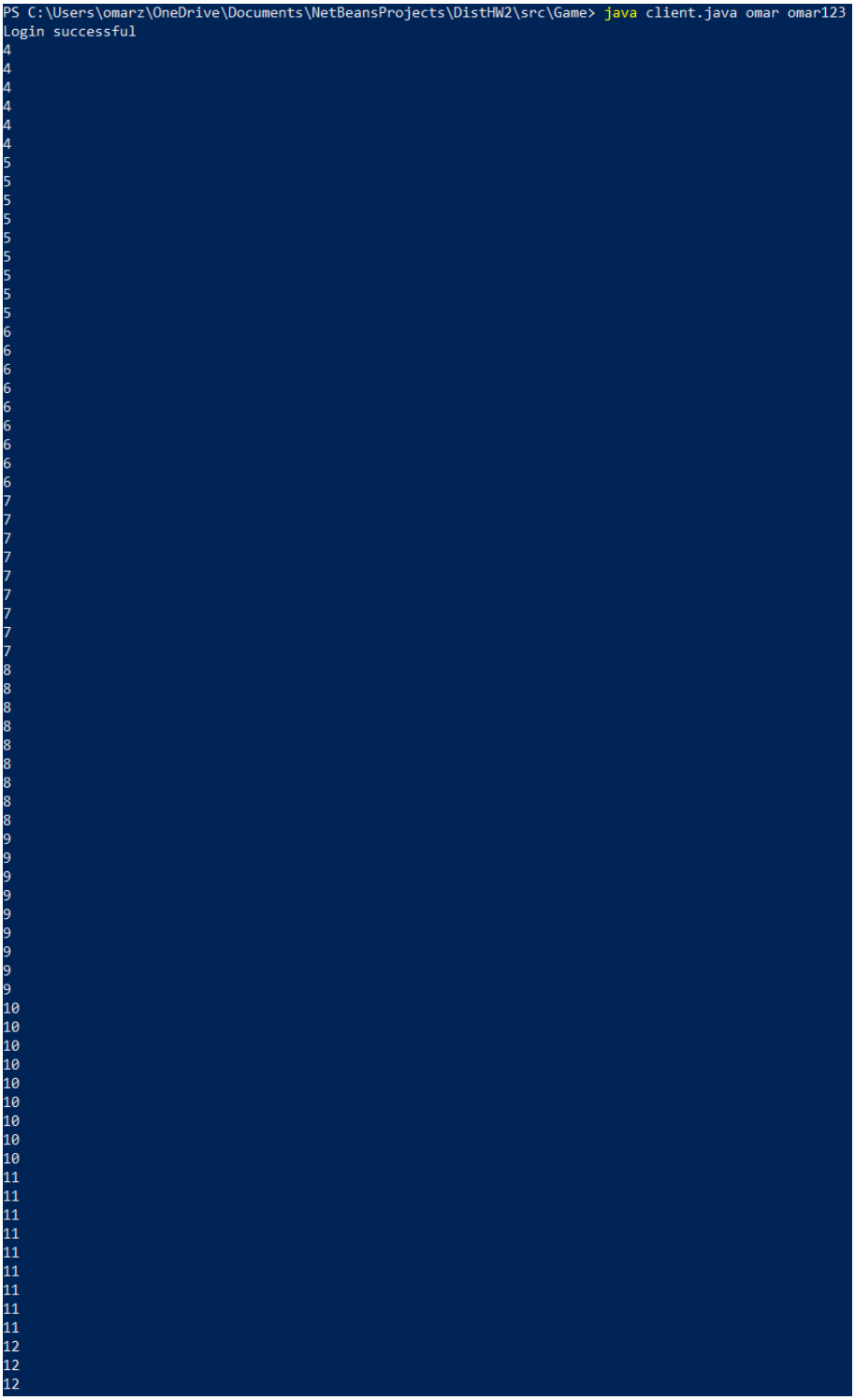
Player.txt:

```
omar/omar123
hadi/hadi123
fares/fares123
mariem/mariemBest
shlash/shlash123
```

Results:

From the following figures, we can see that the results are just as expected, once players connect and are placed in the queue, we can see that each player can see the timer, and once five players joined the game, the game started and went smoothly, even when one of the player was disconnected (meriem), the game kept going until the end.

Omar:



```
12
12
12
12
12
12
12
12
12
12
13
13
13
13
13
13
13
13
13
13
14
14
14
14
14
14
14
14
14
15
15
15
15
15
game is starting
Question for shlash
What is 3 + 7 ?
Correct, good job
[shlash: 1, mariem: 0, omar: 0, fares: 0, hadi: 0]
Question for mariem
What is 4 + 10 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 0, fares: 0, hadi: 0]
Question for omar
Question: What is 8 + 4 ?
Your answer: 12
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 5 + 6 ?
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for hadi
What is 3 + 5 ?
Wrong, The correct answer was: 8
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for shlash
What is 2 + 9 ?
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for mariem
What is 7 + 9 ?
Question for omar
Question: What is 9 + 7 ?
Your answer: 5
```

```
Wrong, The correct answer was: 16
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 6 + 5 ?
Question for hadi
What is 10 + 1 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for shlash
What is 8 + 6 ?
Wrong, The correct answer was: 14
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for mariem
What is 4 + 6 ?
Question for omar
Question: What is 6 + 7 ?
Your answer: 13
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 1]
Question for fares
What is 5 + 3 ?
Question for hadi
What is 1 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 2]
Game over. Final Scores:
shlash: 1
mariem: 1
omar: 2
fares: 0
hadi: 2
Thank you for playing!
```

Fares:

```
PS C:\Users\omarz\OneDrive\Documents\NetBeansProjects\DistHW2\src\Game> java client.java fares fares123
Login successful
9
9
9
9
10
10
10
10
10
10
10
10
10
11
11
11
11
11
11
11
11
11
11
12
12
12
12
12
12
12
12
12
12
12
13
13
13
13
13
13
13
13
13
13
14
14
14
14
14
14
14
14
14
15
15
15
15
15
game is starting
Question for shlash
What is 3 + 7 ?
Correct, good job
[shlash: 1, mariem: 0, omar: 0, fares: 0, hadi: 0]
Question for mariem
What is 4 + 10 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 0, fares: 0, hadi: 0]
Question for omar
What is 8 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
Question: What is 5 + 6 ?
Your answer: 3
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for hadi
```

```
What is 3 + 5 ?
Wrong, The correct answer was: 8
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for shlash
What is 2 + 9 ?
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for mariem
What is 7 + 9 ?
Question for omar
What is 9 + 7 ?
Wrong, The correct answer was: 16
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
Question: What is 6 + 5 ?
Your answer:
You ran out of time, be faster next time.
Question for hadi
What is 10 + 1 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for shlash
What is 8 + 6 ?
Wrong, The correct answer was: 14
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for mariem
What is 4 + 6 ?
Question for omar
What is 6 + 7 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 1]
Question for fares
Question: What is 5 + 3 ?
Your answer:
Invalid input. Numbers only.
Question for hadi
What is 1 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 2]
Game over. Final Scores:
shlash: 1
mariem: 1
omar: 2
fares: 0
hadi: 2
Thank you for playing!
```

Hadi:

```
PS C:\Users\omarz\OneDrive\Documents\NetBeansProjects\DistHW2\src\Game> java client.java hadi hadi123
Login successful
15
game is starting
Question for shlash
What is 3 + 7 ?
Correct, good job
[shlash: 1, mariem: 0, omar: 0, fares: 0, hadi: 0]
Question for mariem
What is 4 + 10 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 0, fares: 0, hadi: 0]
Question for omar
What is 8 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 5 + 6 ?
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for hadi
Question: What is 3 + 5 ?
Your answer: 3
Wrong, The correct answer was: 8
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for shlash
What is 2 + 9 ?
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for mariem
What is 7 + 9 ?
Question for omar
What is 9 + 7 ?
Wrong, The correct answer was: 16
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 6 + 5 ?
Question for hadi
Question: What is 10 + 1 ?
Your answer: 11
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for shlash
What is 8 + 6 ?
Wrong, The correct answer was: 14
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for mariem
What is 4 + 6 ?
Question for omar
What is 6 + 7 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 1]
Question for fares
What is 5 + 3 ?
Question for hadi
Question: What is 1 + 4 ?
Your answer: 5
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 2]
Game over. Final Scores:
shlash: 1
mariem: 1
omar: 2
fares: 0
hadi: 2
Thank you for playing!
```

Mariem:

```
PS C:\Users\omarz\OneDrive\Documents\NetBeansProjects\DistHW2\src\Game> java client.java mariem mariemBest
Login successful
9
9
9
9
9
10
10
10
10
10
10
10
10
10
11
11
11
11
11
11
11
11
11
11
12
12
12
12
12
12
12
12
12
12
12
13
13
13
13
13
13
13
13
13
13
14
14
14
14
14
14
14
14
15
15
15
15
15
game is starting
Question for shlash
What is 3 + 7 ?
Correct, good job
[shlash: 1, mariem: 0, omar: 0, fares: 0, hadi: 0]
Question for mariem
Question: What is 4 + 10 ?
Your answer: 14
Correct, good job
[shlash: 1, mariem: 1, omar: 0, fares: 0, hadi: 0]
Question for omar
What is 8 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 5 + 6 ?
Wrong, The correct answer was: 11
```

```
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for hadi
What is 3 + 5 ?
Wrong, The correct answer was: 8
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for shlash
What is 2 + 9 ?
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for mariem
Question: What is 7 + 9 ?
Your answer:
You ran out of time, be faster next time.
Question for omar
What is 9 + 7 ?
Wrong, The correct answer was: 16
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 6 + 5 ?
Question for hadi
What is 10 + 1 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for shlash
What is 8 + 6 ?
Wrong, The correct answer was: 14
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for mariem
Question: What is 4 + 6 ?
Your answer:
Error readline:Connection reset by peer
```

Shlash:

```
PS C:\Users\omarz\OneDrive\Documents\NetBeansProjects\DistHW2\src\Game> java client.java shlash shlash123
Login successful
9
9
10
10
10
10
10
10
10
10
10
11
11
11
11
11
11
11
11
11
11
12
12
12
12
12
12
12
12
12
12
12
12
13
13
13
13
13
13
13
13
13
13
14
14
14
14
14
14
14
14
14
15
15
15
15
15
game is starting
Question for shlash
Question: What is 3 + 7 ?
Your answer: 10
Correct, good job
[shlash: 1, mariem: 0, omar: 0, fares: 0, hadi: 0]
Question for mariem
What is 4 + 10 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 0, fares: 0, hadi: 0]
Question for omar
What is 8 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 5 + 6 ?
Wrong, The correct answer was: 11
```

```
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for hadi
What is 3 + 5 ?
Wrong, The correct answer was: 8
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for shlash
Question: What is 2 + 9 ?
Your answer: 3
Wrong, The correct answer was: 11
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for mariem
What is 7 + 9 ?
Question for omar
What is 9 + 7 ?
Wrong, The correct answer was: 16
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 0]
Question for fares
What is 6 + 5 ?
Question for hadi
What is 10 + 1 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for shlash
Question: What is 8 + 6 ?
Your answer: 1
Wrong, The correct answer was: 14
[shlash: 1, mariem: 1, omar: 1, fares: 0, hadi: 1]
Question for mariem
What is 4 + 6 ?
Question for omar
What is 6 + 7 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 1]
Question for fares
What is 5 + 3 ?
Question for hadi
What is 1 + 4 ?
Correct, good job
[shlash: 1, mariem: 1, omar: 2, fares: 0, hadi: 2]
Game over. Final Scores:
shlash: 1
mariem: 1
omar: 2
fares: 0
hadi: 2
Thank you for playing!
```

Criticism & Improvements:

There is not much to criticize here other than the UI which is not very user friendly, which can be replaced with something better like a GUI, also only one server is handling the players, which can pose some issues when it comes to scalability.