Project 5 - Team #9

Group Members:

Ashley Santos - asanto3@uic.edu

Ming Chen - mchen217@uic.edu

Eldin Vujic - evujic2@uic.edu

Vishal Patel - vpate48@uic.edu

Project #5 Design Documentation

Section 1 - Project Synopsis and Purpose:

For our final project, our group decided to create a Number Guessing Game, which utilizes multi-threading through Servers and Clients, as well as elements of JavaFX. The premise of the project is that four clients or players must connect to a server, in which that the server will generate a random number between the integers of 0 and 999. Following, each player is allowed five guesses. Players earn points based on the correctness of their guess and the point system is dependent on which number guess was the correct one.

For example, if a player guesses the correct number on their first guess, they will earn 10 points. The point system continues that if a player guesses the exact number on their second guess, they will earn 4 points, which is the number of guesses they had left remaining. Continuing, if a player guesses the exact number on their third guess, they will earn 3 points. This breakdown continues until the player is on their final guess which if they guess correctly, the player will earn one point. However, if all of the player's guesses are incorrect, then the player does not earn any points.

There is no set winner of the game; the goal of the game is to have the highest score among all the players within the game. The player is able to leave the game at any point; however, the other clients must wait for another client to enter the server, thus totaling four players in order for the game to start once again.

Section 2 - High-level Entities:

Within the Server Folder:

1. Server Class: This class contains the functionality that checks whether or not if it is a Server, and has getter method that returns both the IP and Ports.
2. Controller: Receives commands based on user entry, and sends the commands to the database, and updates the view of the server.
3. Main Class: Handles the GUI of the Server
4. Network Connection File:
    a. Network Connection Class
    b. Connection Thread Class
    c. Client Thread Class
    d. Player Class
    e. Game Play Class

Within the Client Folder:

1. Client Class: This class contains the functionality that checks whether or not if it is a Server, and has getter method that returns both the IP and Ports.

2. Controller: Receives commands based on user entry, and sends the commands to the database, and updates the view of the client.
3. Main Class: Handles the GUI of the Client
4. Network Connection File:
    a. Network Connection Class
    b. Connection Thread Class

Section 3 - Low-level Entities:

Game Play Class

1. generateNumber(): This method generates the server's random number.
2. howClose(int guess): This method determines whether the player's current guess is too high or too low.
3. winOrLose(int guess): This method determines whether or not the player's guess

This class is used to control the gameplay of the game. This class has a method that generates a random number for the players to guess. When a player guesses a guess it will return if they win or how close they are to the guess. This is in the server class.

Player Class

1. setName(String i): This method sets the name of the player within the game.
2. returnName(): This method returns the name of the player in the game.
3. getPlayerData(): This method gets the player data.
4. setPlayerData(String s): This method sets the player data.
5. getWon(): This method gets the winner of the game.

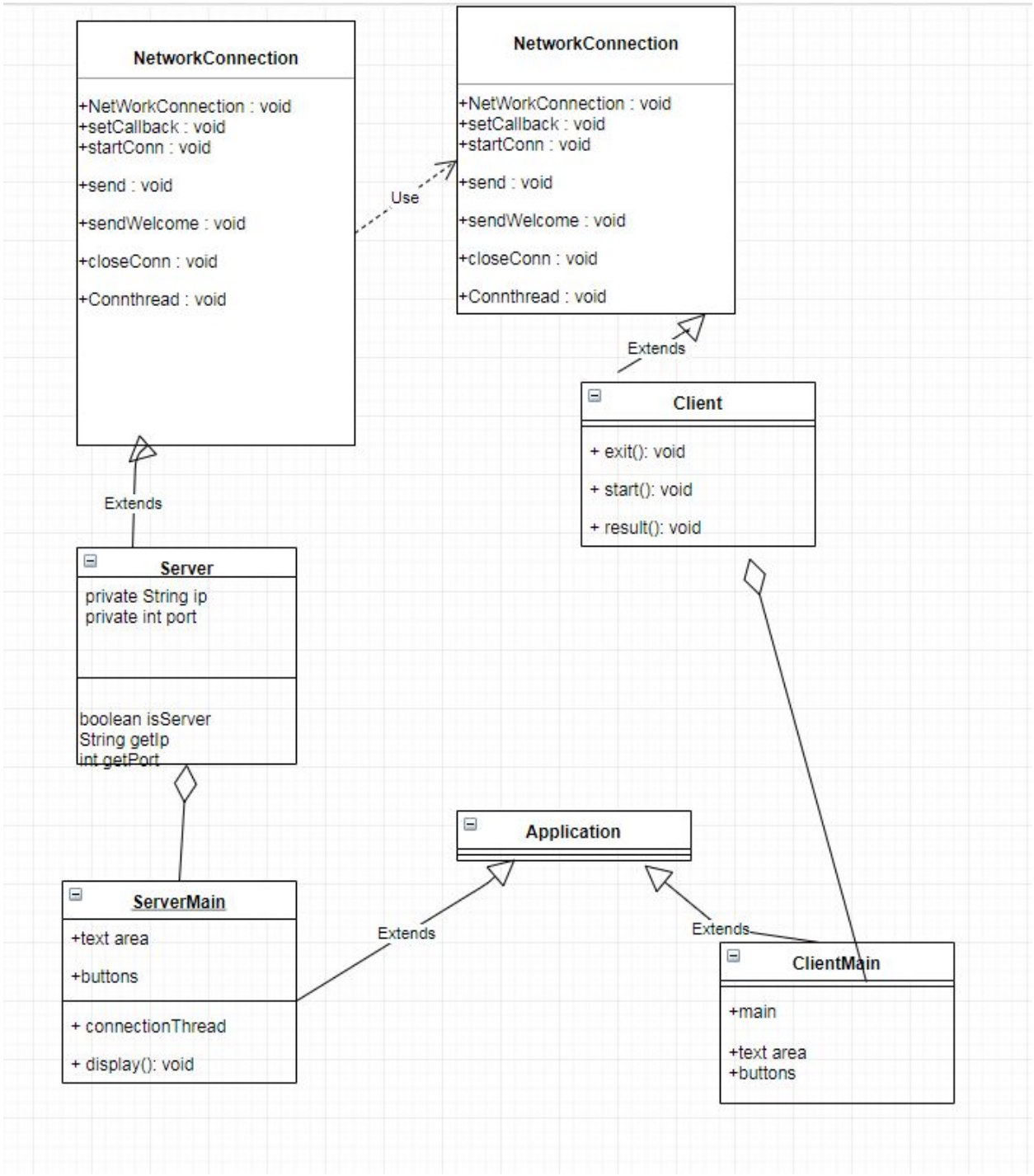
6. setWon(boolean w): This method sets the winner of the game.
7. getGuesses(): This method gets how many guesses the player has.
8. setGuesses(int g) This method sets the number of guesses the player has.
9. deleteGuess(): This method decrements the player's guesses count by one based on how many guesses were made during the game.
10. getPlayerPoints(): This method returns how many points the player has.
11. setPlayerPoints(int p): This method sets the players points.
12. getClientStatus(): This method gets the client status.
13. updateClientStatus(): This method updates the client status.
14. resetPoints(): This method resets the player's points.
15. setId(int i): This method sets the player's identification number.
16. returnId(): This method returns the player's identification number.

This class is made to keep the statues of the player. It keeps the points of the player. Then it keeps track of how many guess the player has made and whether or not the player has guessed. Also, keeps track of the id of the player so we know who the player is. Server Class.

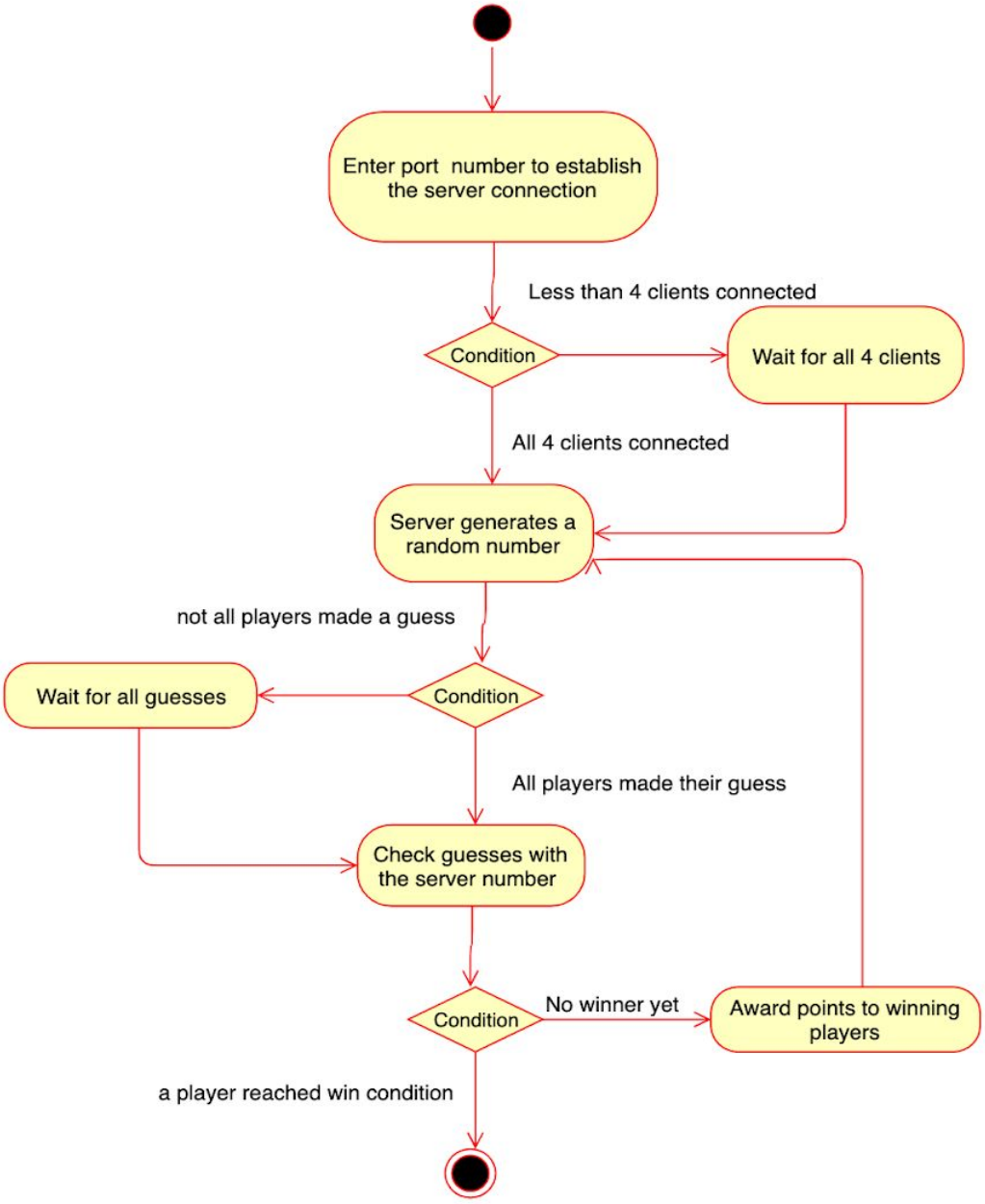
The network connection file has many thread and connection classes to connect to clients, and receives and sends data to the client for gameplay purposes. It waits for clients to connects then begins the game process.

Client code deals with what info the client has to send for the game such as inputs which are the guess to the game.

Model (UML Diagram):



Interaction (Activity Diagram):



Section 4 - Benefits, Assumptions, Risks & Issues:

Benefits:

1. Practice making server and client code
2. Practice multithreading
3. Able to connect gui components into a server client game
4. The program that we design could be a base for an app or software game
5. Good experience with working with a team

Issues:

1. We had some trouble at first with the chat part
2. We had some trouble with how the random number was being generate it was crashing the server
3. We didn't like how we first created the game so we decided to give each player five guesses and they get more points for the least amount of guess