**Hanoi University of Science and Technology  
School of Information and Communication Technology**



**PROJECT REPORT**

**Project name: Tic Tac Toe Game with Socket Programming**

**Week II**

***Subject: Network Programming***

**Instructor: Ph.D Truong Dieu Linh**

**Group 3:**

|  |  |  |
| --- | --- | --- |
| **Student ID** | **Student name** | **Class** |
| 20176732 | **Tran Thai Duong** |  |
| 20176845 | **Nguyen Manh Phuc** |  |
| 20184238 | **Nguyen Thi Minh Chau** | **ICT 01 – K63** |

**Hanoi, 2021**

PART I: Application model

The application uses asynchronous socket in order to create connection and communicate between clients and server. In this model, client or server makes the non-blocking call and start other background tasks instead of waiting for the request to be completed. When the operation is done, a completion callback is generated to finish the I/O operation.

Asynchronous socket communication brings several advantages:

- Allow multiple connection at the same time with good quality: while blocking (in blocking socket calls) prevents the program from completing other operation until the request is returned, and polling (in selecting) consumes computer memory and decrease performance, asynchronous socket allows background works to be continue when the I/O call is being processed, and only notifies when the call is completed.

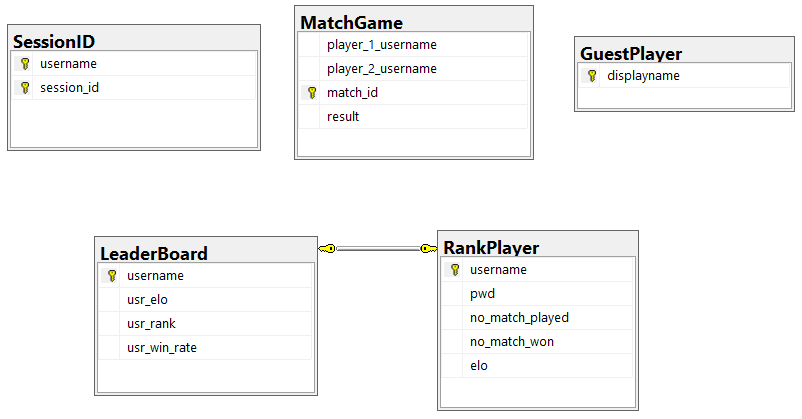
- Save resources and avoid threading problems: Although creating threads for handling multiple connection at the same time also allow many clients to communicate with the server, this waste the memory since each thread need its own space for saving its metadata. Also, because threads use the same memory space of the program and separated threads may have operation on the same resource at the same time, deadlocks may raise. Asynchronous socket can be implemented on one thread only, which does not require additional memory, prevent mutexes and other synchronizing problems of multithreaded programs.

Clients and server communicate using TCP protocol. TCP has mechanisms for guaranteeing transmission of packets, so that the program can be eliminated from having to check if the message is successfully sent. The data stream is not complicated and there is no requirement as for streaming services; therefore, TCP is considered suitable.

Since there is the need of keeping track of each action performed on the client side, the communication is stateful, and information of the client is saved in a database on the server side for further references.

Part II: Databases & Libraries

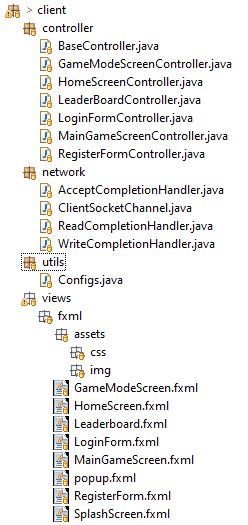
In this program, we use SQLite to save database, with the diagram as shown

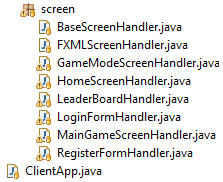


In order to create message, library org.json is used

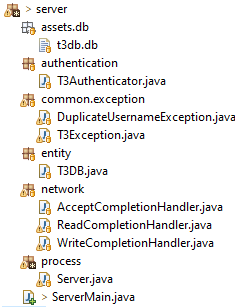
PART III: Structure of the project – Functions and methods for processing data

On the client side

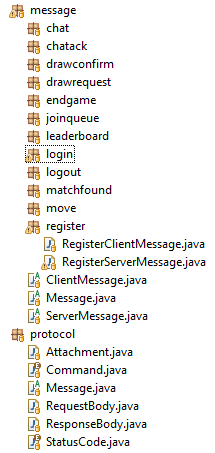




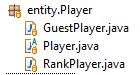
On the server side



Classes for protocol and message

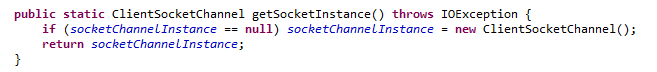


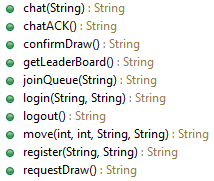
Classes for managing players

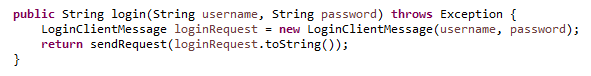


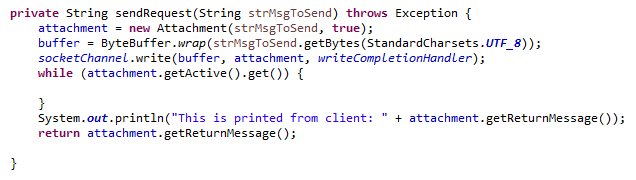
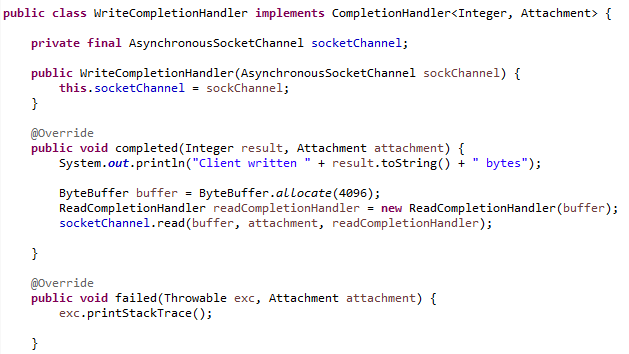
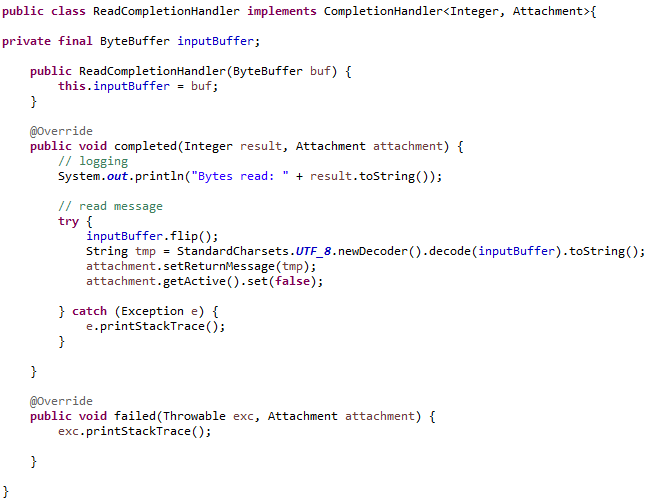
PART IV: Code implementation for network

On the client side



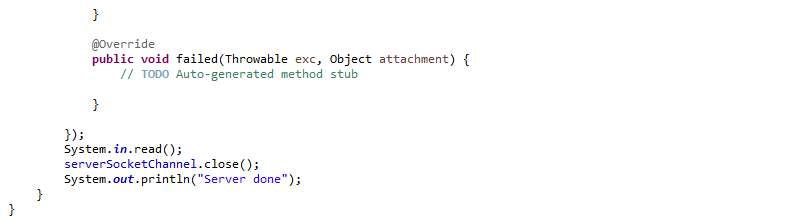


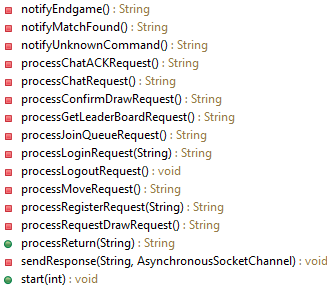


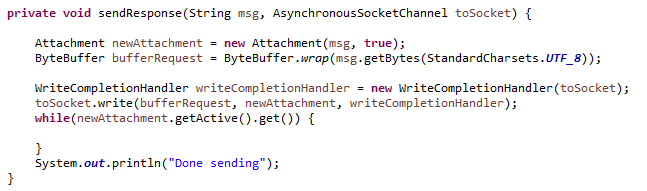
  

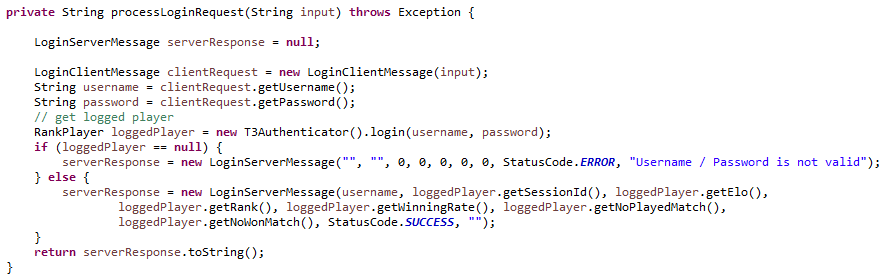
On the server side

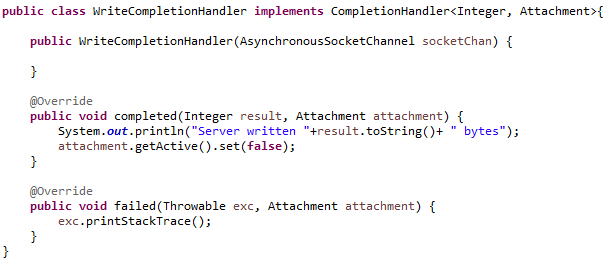










PART V: Integration of network programming to the application & demo

