

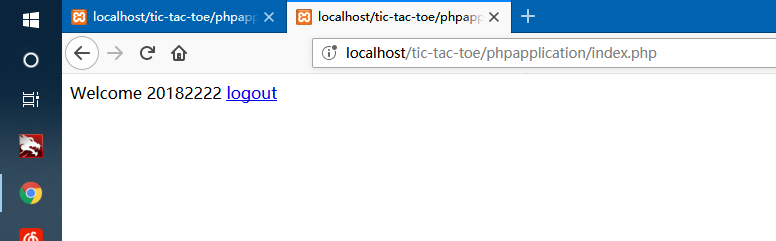
Department of Electronic & Computer Engineering

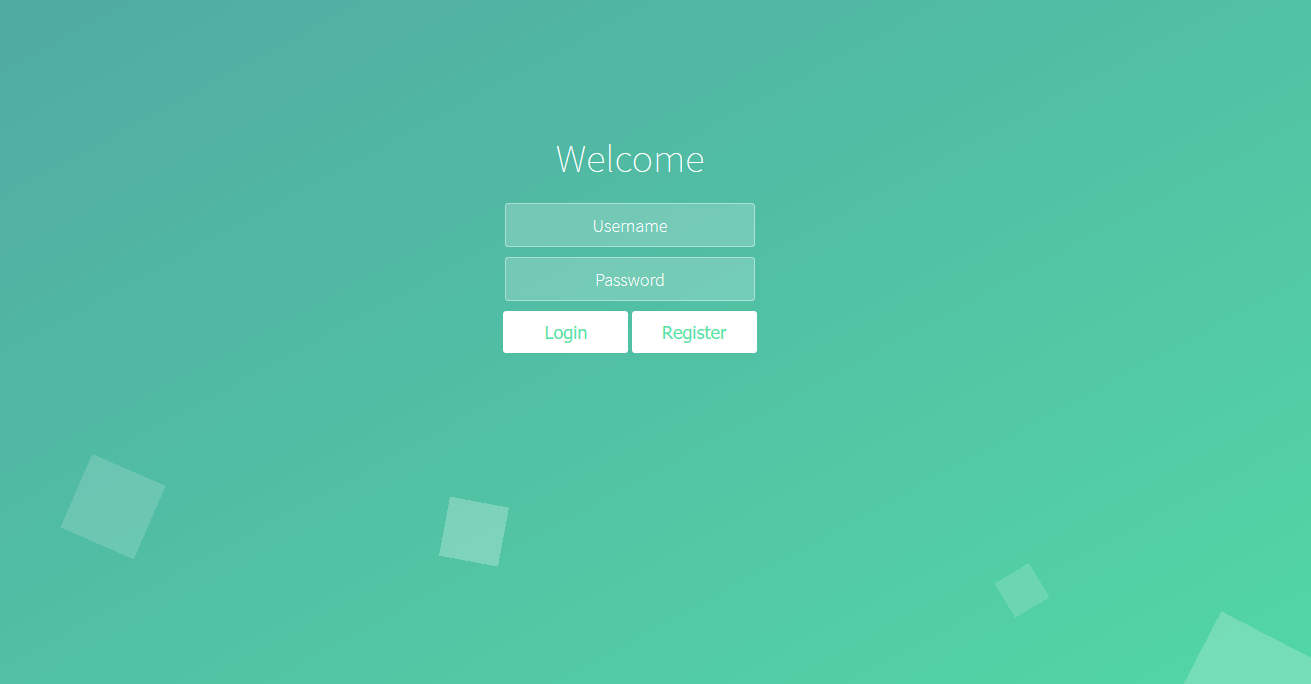
EE4023 - DISTRIBUTED SYSTEM

Tic - Tac - Toe Team Project

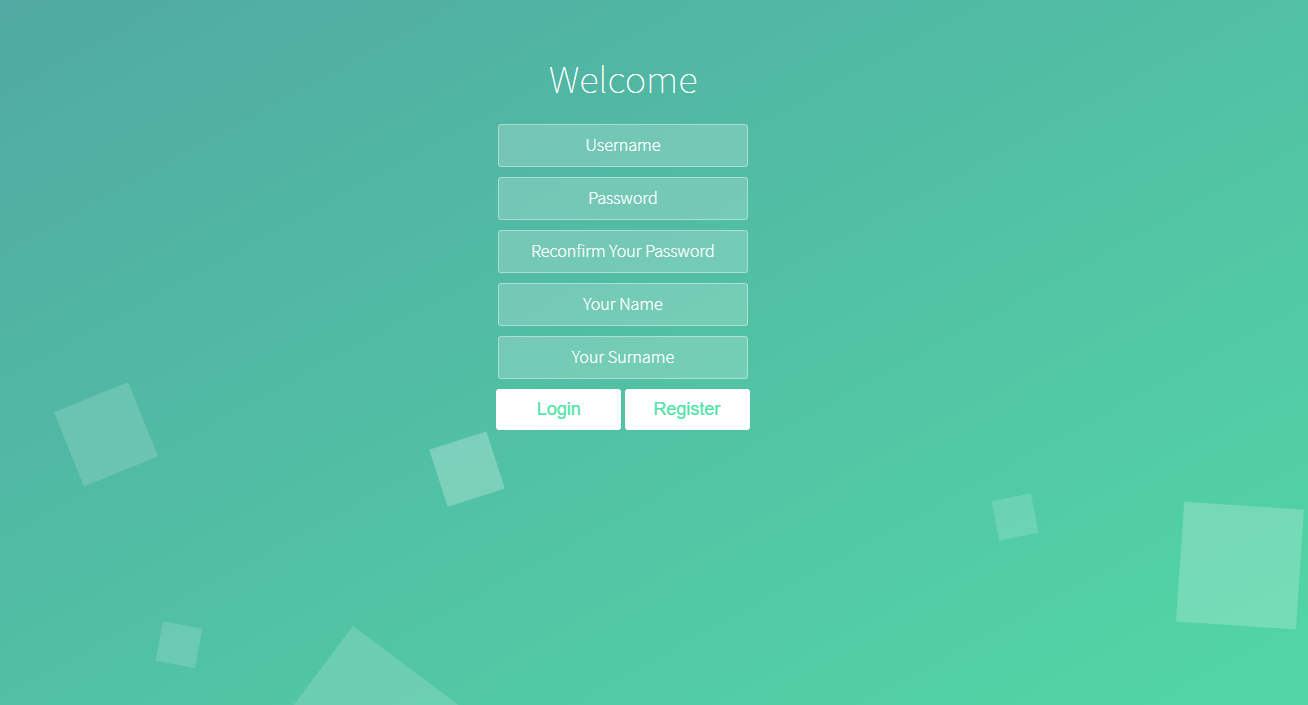
|  |  |
| --- | --- |
| Team Member: | Yiran Gao 17083214 |
|  | Xinting Li 17079306 |
| Team Name: | OvO |
| Course: | BSc in Computer Systems |
| Academic Year: | 2018/11 |
|  |  |

**“Submitted in partial fulfilment for the requirements of the Bachelor of Science Degree in the University of Limerick”**

* **PHP System - Structure Yiran Gao**
  + Project Folder
    - tic-tac-toe
      * phpapplication
        + allopengames.php
        + board.php
        + connection.php
        + deletegame.php
        + gamepage.html
        + gamepage.php
        + gamepageonp2.html
        + gamepageonp2.php
        + index.php
        + joingame.php
        + joingamelist.html
        + judge.php
        + leaderboard.html
        + leaderboard.php
        + login.html
        + login.php
        + main.html
        + main.php
        + myopengame.php
        + newgame.php
        + register.html
        + register.php
        + scoresystem.html
        + scoresystem.php
        + surrender.php
* **PHP System - Basic Logic**
  + In order to implement tic tac toe in php based system, I developed html and js files as the front end to build the GUI to players and using php languages to connect web server to fetch data and forms then provided them to the html page.
  + I follow the structure of binding both html page and php file and also provide individual php file as extended function in the game system.
  + Tools: MySQL Sublime NetBeans
* **PHP System - Implementation**
  + A main page choosing login & register in index.html
  + 
  + A login screen to enter credentials.



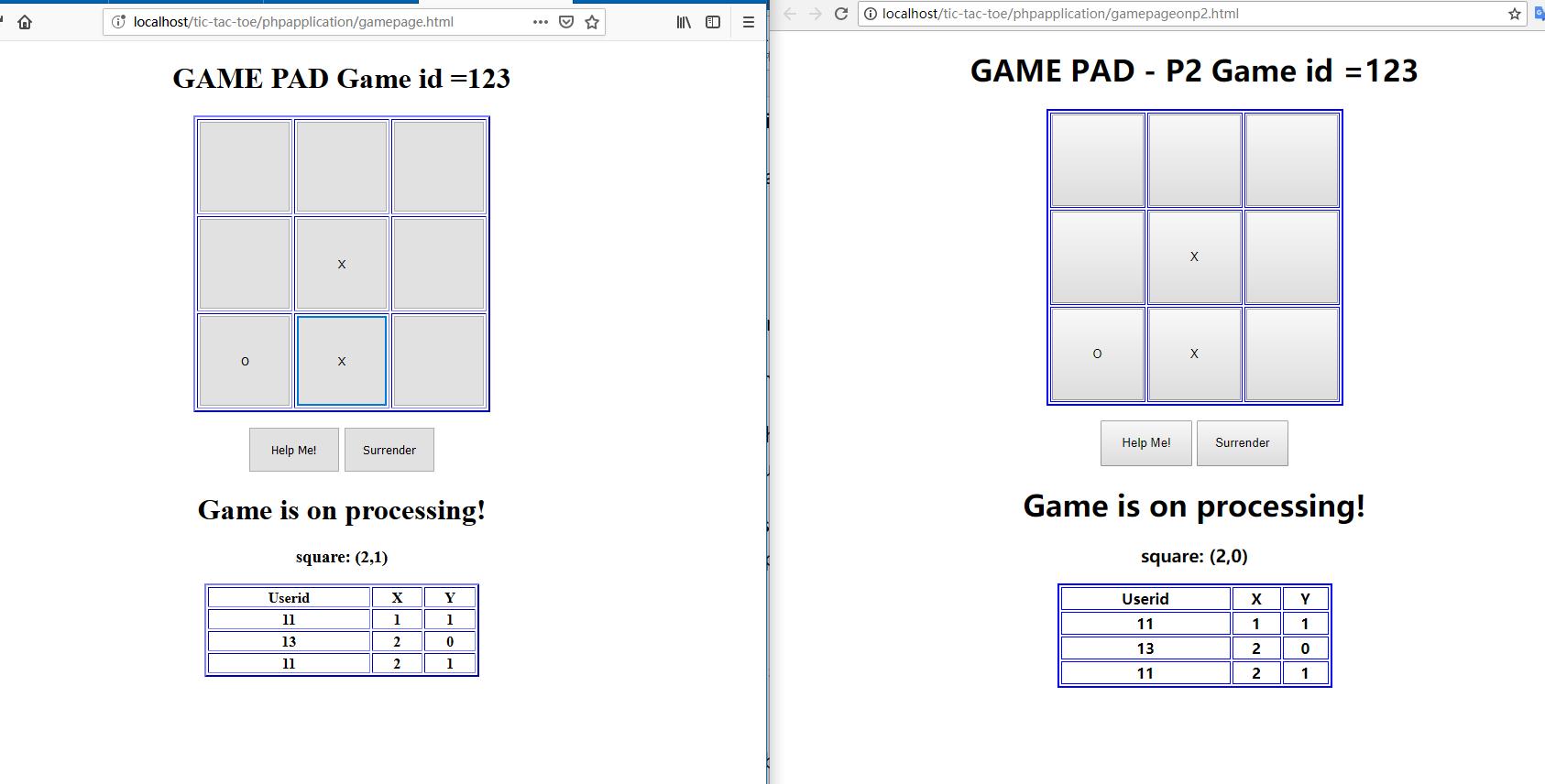
* + A registration screen to create a new account.



* A main screen



* Game panel in P1 & P2



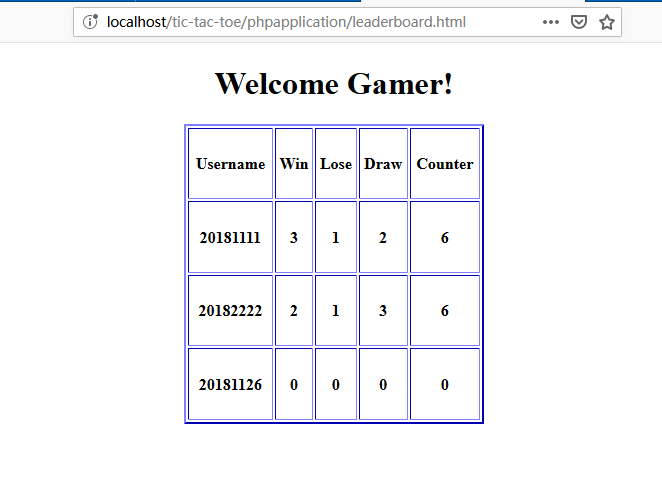
* Join game page



* Score system page



* Rank list (Using thread to fresh table every second)



Java desktop application

Programmer: Xinting Li

Student ID: 17079306

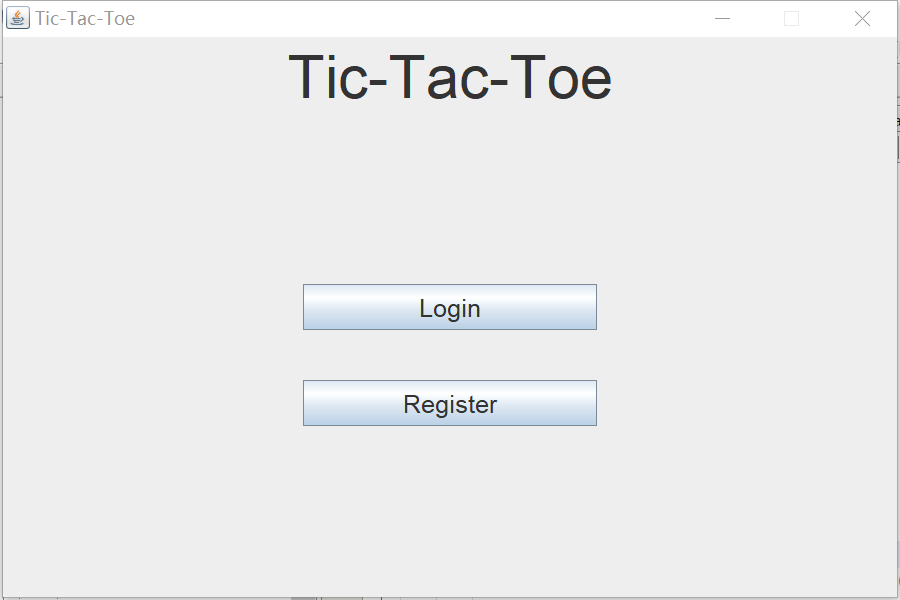
**Overview**

The project is Tic-Tac-Toe game. It is a two players game, X and O. In this game, there is a board with 3 x 3 squares. The goal of Tic-Tac-Toe is to be one of the players to get three same symbols in a row - horizontally, vertically or diagonally - on a 3 x 3 grid.

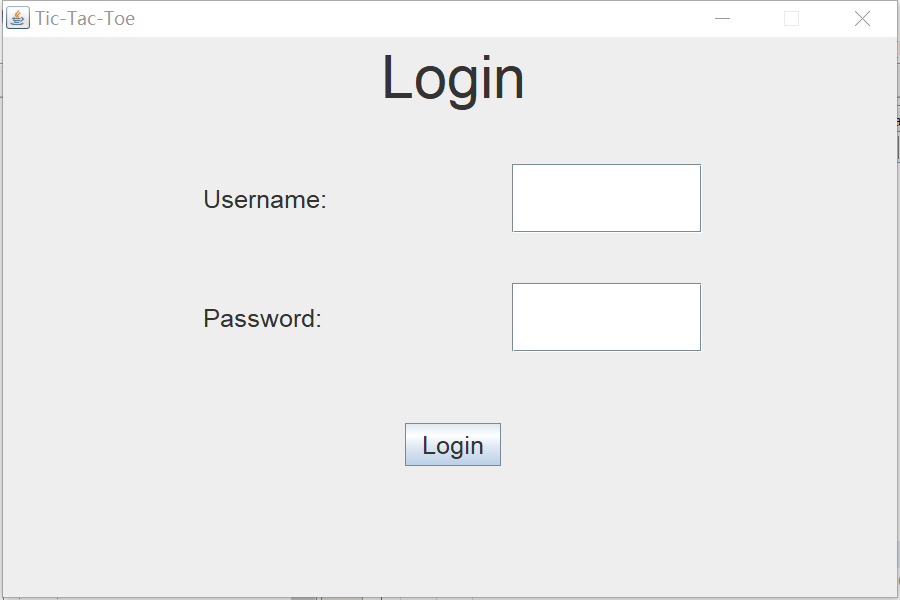
**Design**

Here are the core steps in Java desktop application:

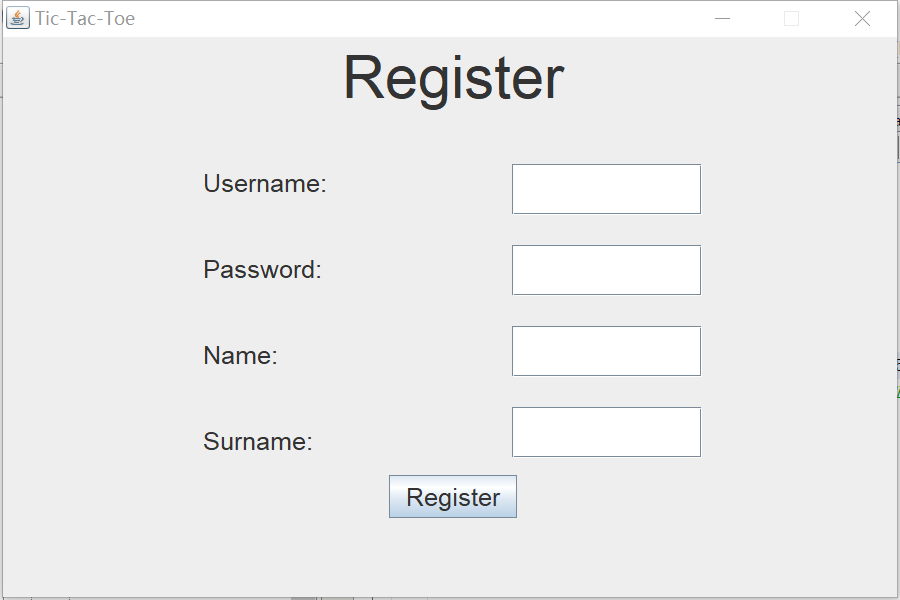
1. Welcome page will be shown. There will been two button, login and register.



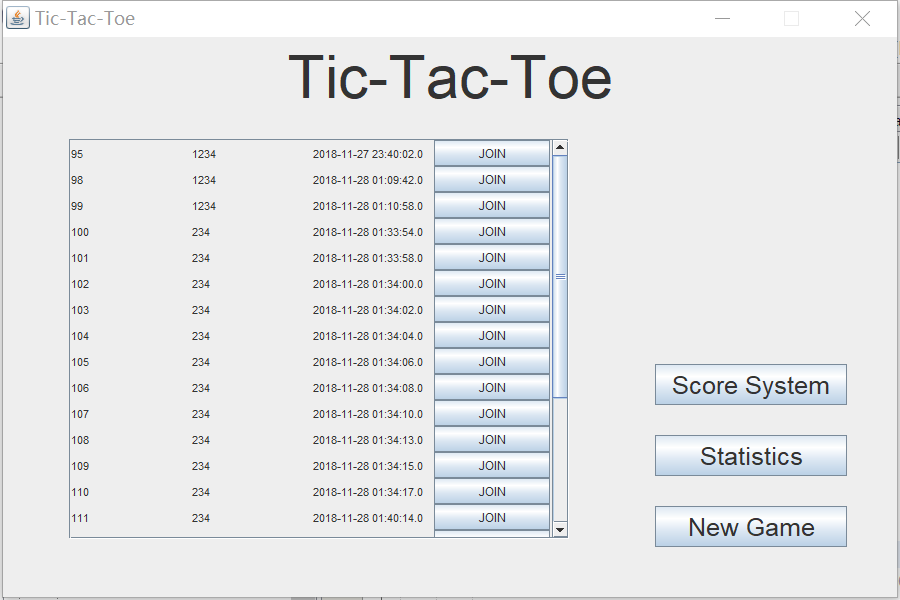
1. Login page: sign in a user



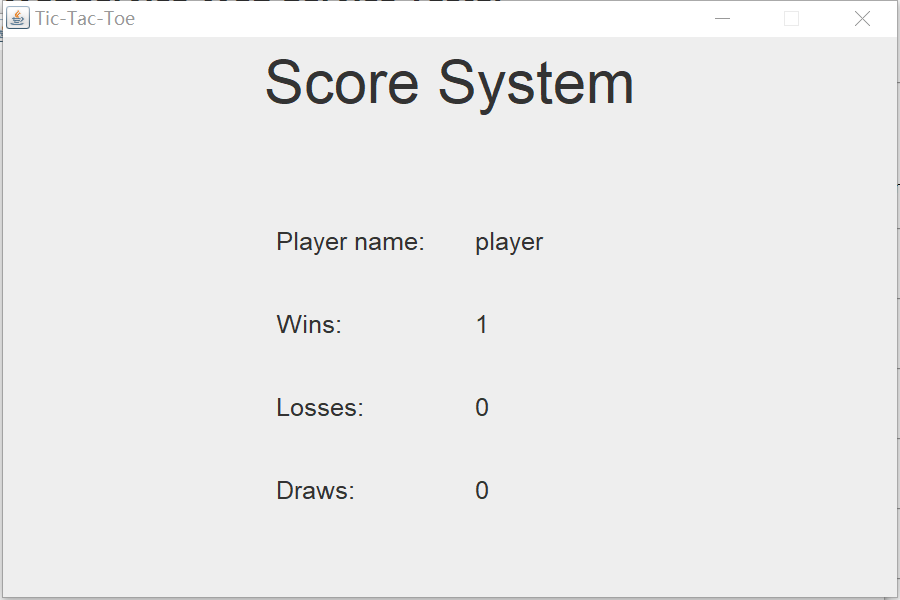
1. Register page: register user



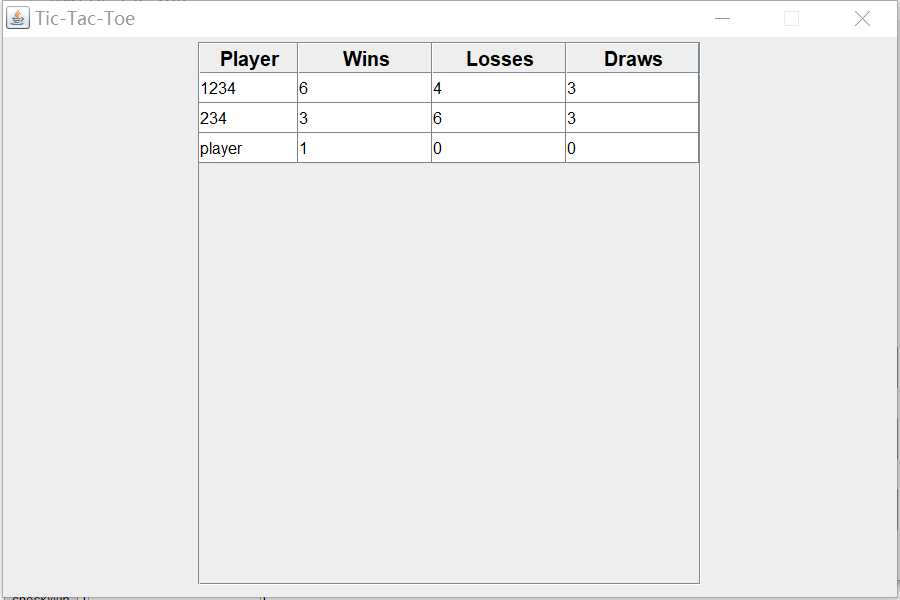
1. After login and register, the main screen of this application will be shown. There will be a Score system button, a Statistic button, a new game button and a table for showing the games which are waiting for a second player to join. The table will be update automatically.



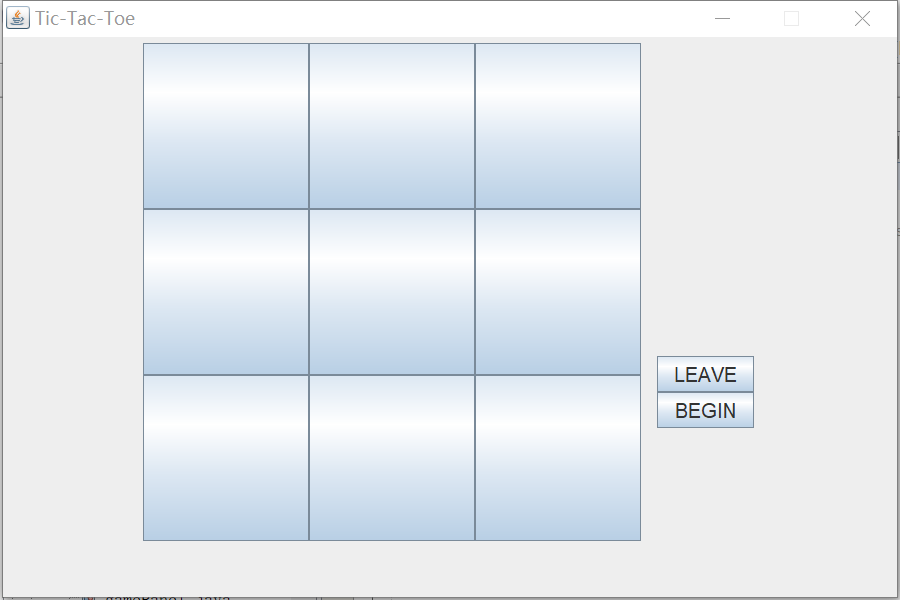
1. Score system: showing the number of wins, losses and draws the player has.



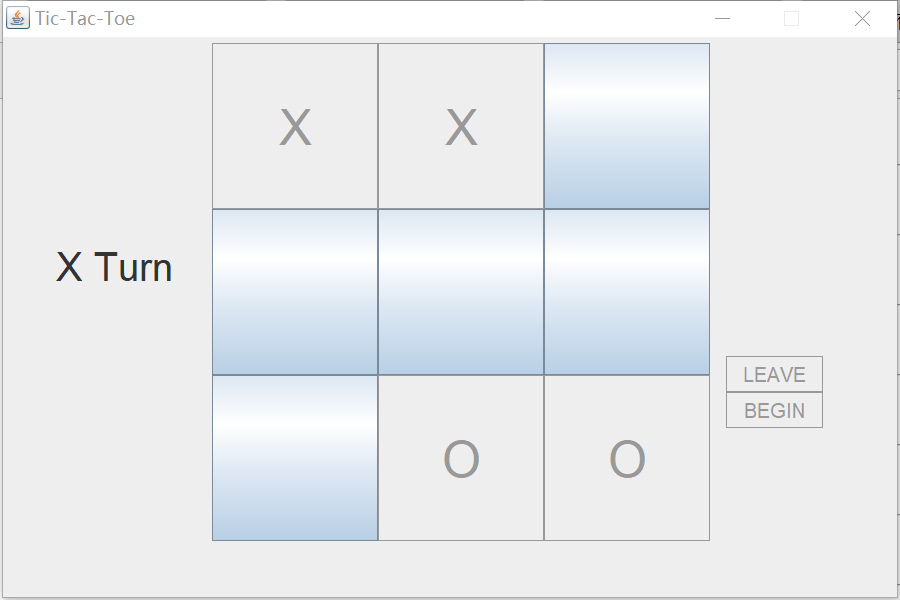
1. Statistic: showing the statistics (wins, losses and draws) for each registered user

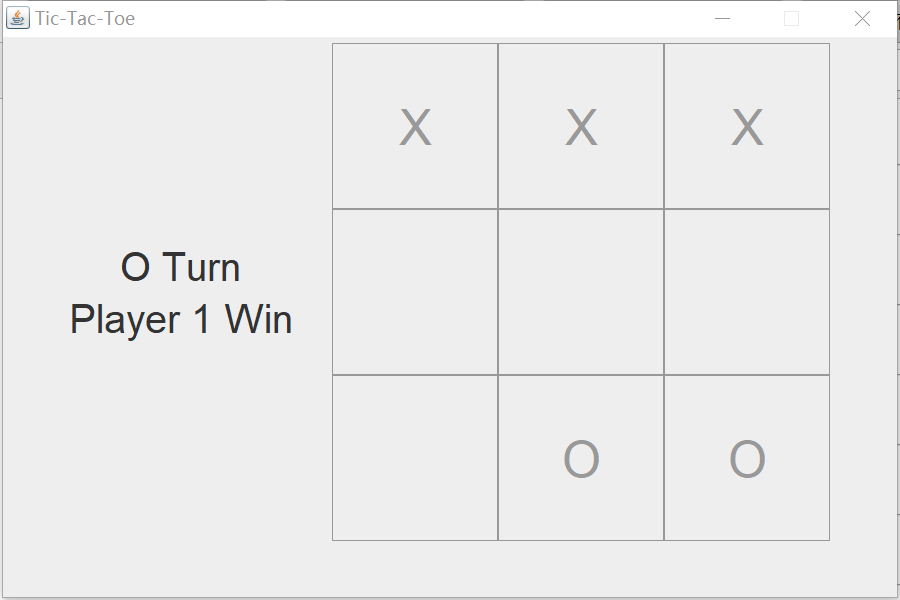


1. New game: after click the new game button, the game page will be shown. There is a leave button, a begin button and a game table on this page.
   1. If click the leave button, it will return to the main screen and the game will be deleted.



* 1. If not click the leave button and the second player has joined the game, then click begin button, the game page will be shown and player could begin to play the game. When the game is processing, the leave button and begin button cannot be click.

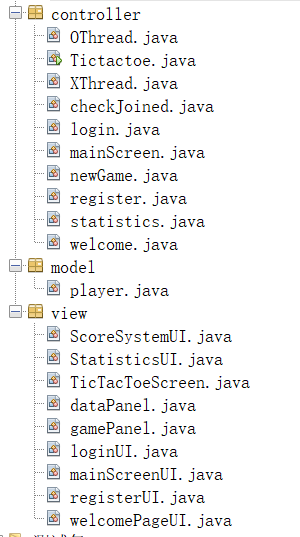




**Implementation**

**Classes and packages:**

There are 22 classes in java part in this project and they are divided into three packages, controller, model and view. The business logic and UI page are store in controller and view respectively.



**Tool:**

* NetBeans
* MySQL
* Apache

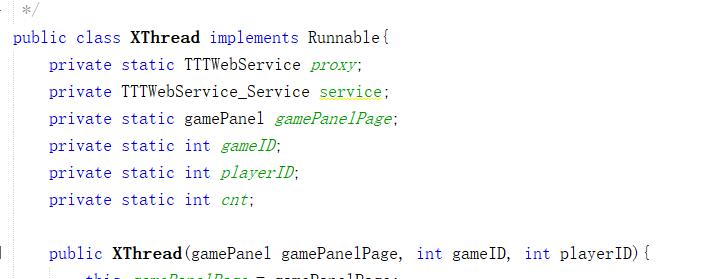
**Code:**

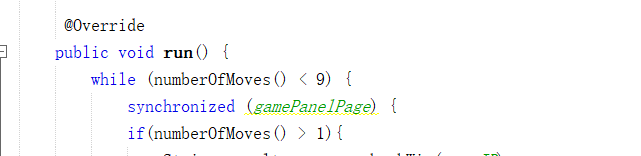
* UI and business logic:

Both of them are developed in java

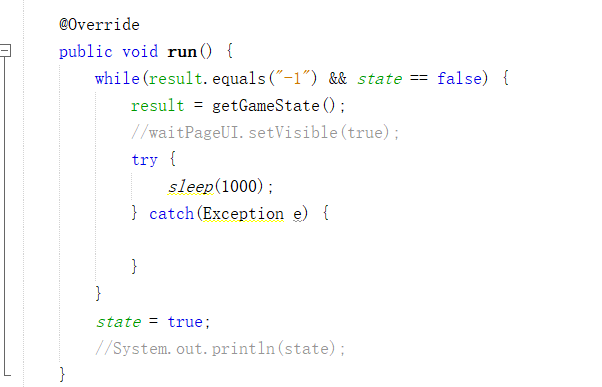
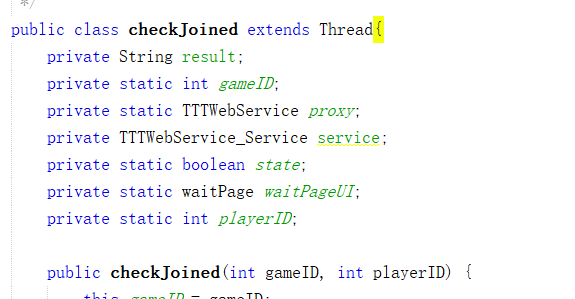
* Thread: is used in three parts:  
  1. when playing games, using thread and synchronized to make sure to know where another player clicked and to control who can take a turn at a time.







2. when waiting for the second player to join.



3. update the table which showing a list of open games that player can join.



**Test:**

Using TTTWebService Web Service Tester to test if could be run successfully.

**Limitations:**

GUI is not so attractive.

We cannot change UI in other thread except UI thread.