Connect 4 Design Document

Table of Contents:

Benchmarks To Meet:

Front End	● Login Page: ○ User authentication: Verify users through correct combination of username and password. ○ Register account ● Homepage: ○ Display token customization, user functionality ● Board Page: ○ Display a visually appealing checker board for players to interact/play the game on. ● Victory and Defeat Page ○ Display corresponding page to each player according to result of game.
Back End	● User Verification ○ Needs to check for correct combination of username and password. ○ Prompt user to create account if they are a new player. ● Gameboard Logic: ○ Connect 4 game rules/logic must uphold during the game ○ Correctly determine winner, loser, or stalemate ● Multiplayer system: ○ Correctly output status of player (“busy”, “available”, “sleeping”). ○ Properly connect two players together
Testing	● Create > 10 mock users, simulate user authentication. ● Simulate player connection and game play.

Estimated Hours

- Token Class (1 hour + skin time)
    - Appearance
    - User it belongs to

- GameBoard Class (3 hour) -
    - Lock and condition represents turns
    - Private List<List<Token>> board = new CopyOnWriteArrayList<List<Token>>()
    - Method to check if valid place to place token and on which row the token lands
        - Int isValid() returns lowest valid row
    - Checking for four in a row (winner) method
        - Prints winner message
        - Ends game
    - Update GameBoard method

- Player Class (extends Thread) (8 hours)
    - Registered boolean
        - Registered players have usernames & can search/be searched
        - boolean, 0 = unregistered, 1 = registered
    - Preferred token variable (default for unregistered players)
        - Akin to private int ID = 10;
    - Boolean insert(int col) [Can't insert -> return false -> prompt new col]
        - Get lock (for gameboard)
        - Calls "isValid()" in GameBoard
        - Set value of GameBoard[isValid()][col] = true;
        - Call Update GameBoard
        - Release lock
    - Boolean to represent if the player's already in a game or not
        - isPlaying
        - Called in server or clientmain class, checks if another player can request to play with this playerw
    - **run()**: while(true) {
        - If (condition notified)
            - Prompt insert where: insert(scanner.getLine()) //check if its int
                - Scanner for now, but should be front end
            - if(winner) { break;}

- ServerMain (4 hours)- Saving login data (stores all usernames and passwords)
    - public bool createGuest()
        - Returns false if any error occurs

    - public bool createPlayer(String user, String pass)
        - Returns false if any error occurs
    - findPlayer()
        - Locates registered player to connect with and forms connection if they're available

- ClientMain [Scanner scanner = new Scanner(System.in)] (5 hours)
    - Pulls data from player class and allows individual user to login to the server to play
    - Contact player system
        - Player inputs which player they want to play with
        - if(isPlaying){ System.out.println("Player is busy.") }

- Front end (10 hours)
    - Login page: input fields for user and pass, create an account, or enter as guest
    - Home page: start search (is the only guest functionality), invite player, token skins (, board skins?), favorite token (and board?)
    - Board page: where you actually play the game
    - Victory/Defeat page: displays after win condition for game reached (player earns in-game currency after each victory, currency can be spent to unlock new skins; daily quest/challenges (store variable in database to track current day)

Test files:
Set up users (user setup, search function), anonymous users test, insert to full token stack test

High Level Requirements:

Summary:
6x7 board layout, with possibly extra board sizes, search feature, account saving and multiplayer. The objective is to place four consecutive tokens, vertically, horizontally, or diagonally. Authenticated users will be able to play with another user of their choice and will have access to special connect-four tokens they can use while playing. Unauthenticated users will only be able to play with random users, and will only be able to use standard red and yellow connect-four tokens.

Landing Page:
The landing page will be the "main menu" for the game and display it's title and a button prompting the user to start playing. The only functionality present on this page will be the play button, which will begin the game upon being pressed.

Login Page:
The login page will present the user with two options: the ability to play as a guest or login. If the user chooses to play as a guest, they will be taken to the guest page and given access to the game itself with restricted functionality. Basically, guests will NOT have the ability to search for specific opponents or add friends, and only be able to queue for a match with random opponents.

If the player chooses to login, they will be taken to a login page. This page will feature two empty boxes where they can enter a string of characters for their username and password. Upon doing so, the program will run a check against our SQL database to see if the username and password belong to an existing user. If so, the user will be logged in. If not, the program will prompt them to enter their credentials again.

Registered User Page:
Once successfully logged in, the user will be taken to the registered user page. This page will give them the option to add friends, play against already existing friends, or play a random opponent.

- Add Friends: If this option is selected, the user can enter the username of a friend they'd like to add. If another user with a matching username is found, the program will save that user's data to the data structure containing friends within the main user.
- If "play friends" is selected, the user will be taken to the friends list where they can select one of their friends to play with.
- If "play random opponent" is selected, the game will randomly match the user with any other users currently online and not already within a game

Add Friends Page:
The add friends page will display a list of the other users the main user is currently friends with. If the "add friends" button is selected, the user will be taken to the "search friends" page, where they can enter the username of the player they are trying to friend.

Search Friends Page:
This page will give the user the ability to search for another user by their username. If a matching registered user is found within the SQL database, they will be saved as one of the current user's friends.

Play Friends Page:
The program will parse through the user's "friends" data structure and display any of the user's friends that are currently online on-screen.

Accept/Decline Page:

If someone matches with a user while they are not currently within a match, the program will take the user to the accept/decline page. Here, the user can choose the “accept” button to have the server connect them with the other user and begin a match. This will take them to the board page to play.

If the user selects “decline”, they will be taken back to whatever page they were previously on.

Board Page:
The board page is where the main game will take place. On this page the game board will be displayed and the player will be able to place pieces within it on their turn.

If the player selects the “leave” button at the top left, they will be taken to the leave page.

Leave Page:
Here, the player will be prompted on whether or not they would like to leave the current match. If the player presses the “leave” button, the server will disconnect them from the other player and they will be taken back to either the registered user page or login page depending on what type of player they are. The other player involved in the match will be notified that their opponent left and taken back to the necessary page in the same fashion.

On the contrary, if the player presses the “stay” button, they will be returned to the board page.

Finished Game Page:
Once a game is completed, the player will be taken to the “finished game” page. Here, their victory or defeat status will be displayed accordingly. If the player presses the “play again” button, the program will check to ensure their opponent selected the “play again” button as well and queue the two players for another match if so.

On the contrary, if either one of the players selects the “leave” button, both players will be returned to the registered user page or login page depending on the type of player they are.

Page Layouts:

Landing page:

Login Page:

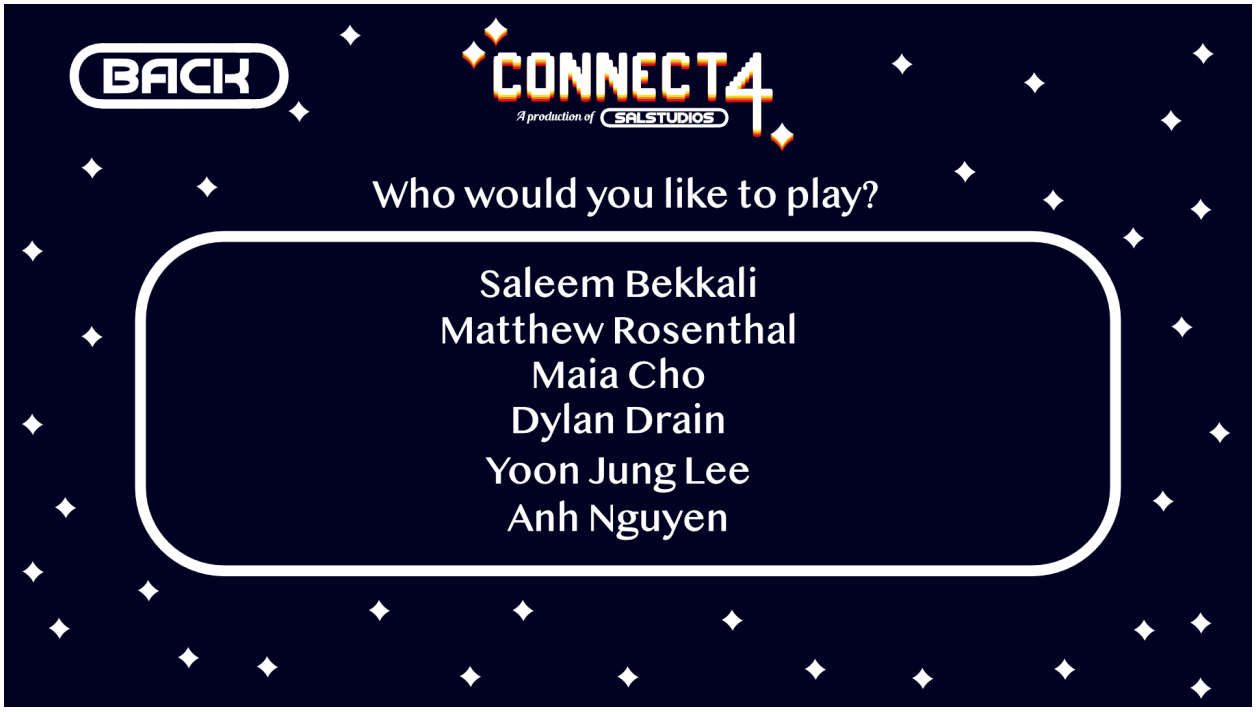
Guest Page:

Registered User Page:

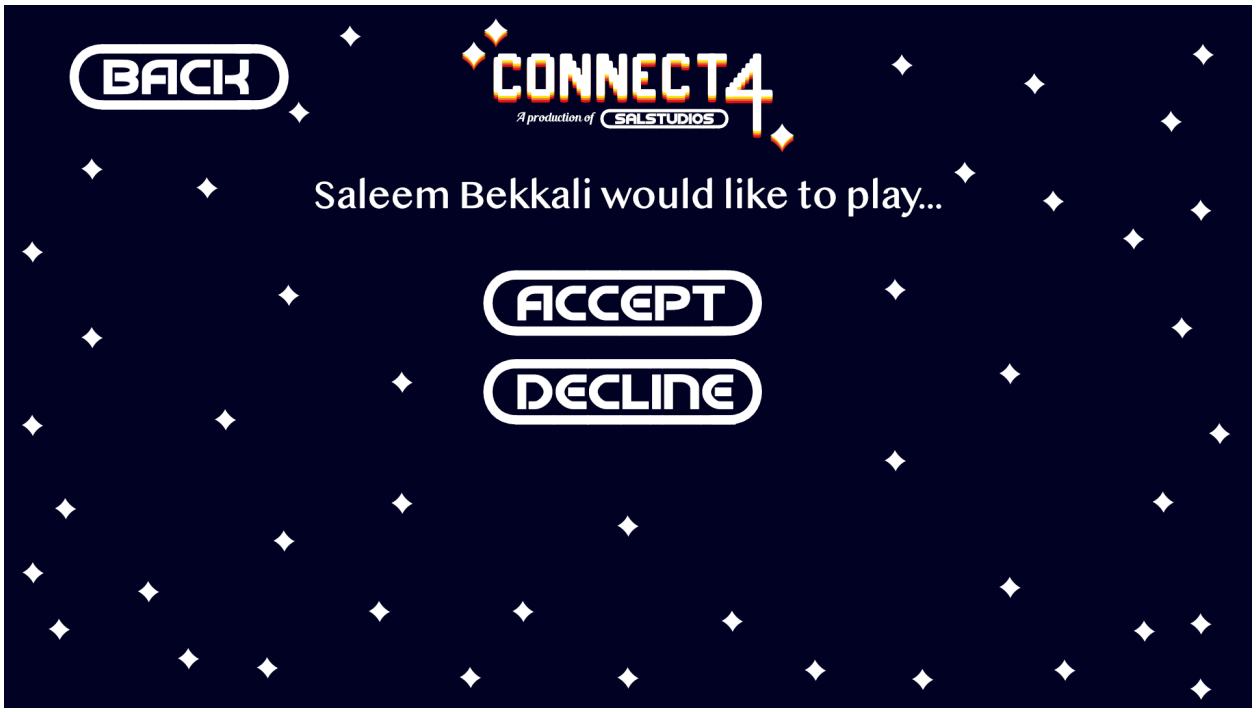
Add Friends Page:

Search Friends Page:

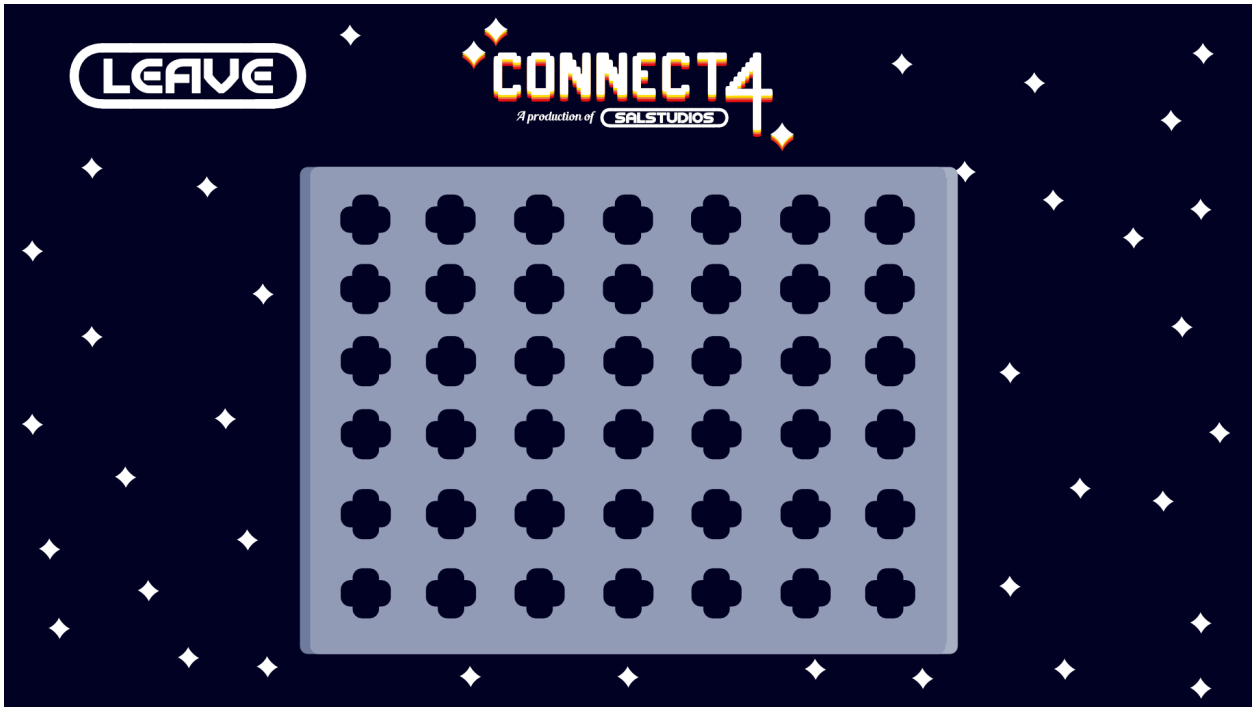
Play Friends Page:



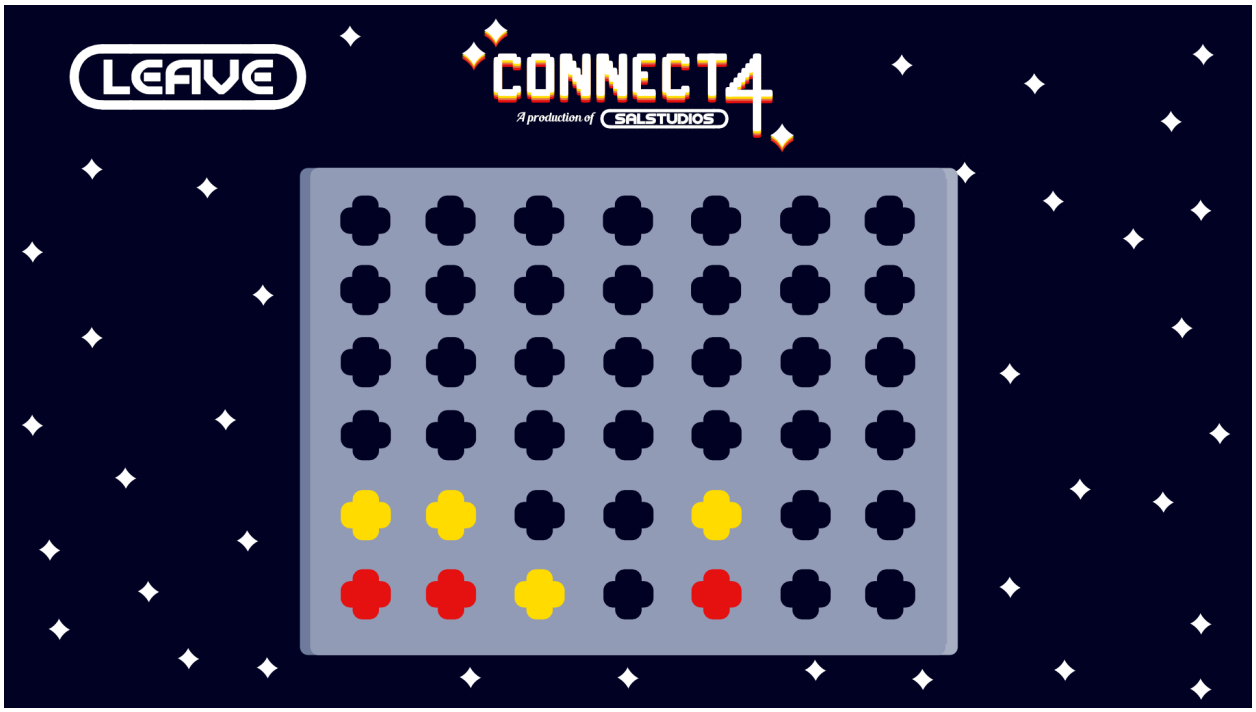
Accept/Decline Notification Page:



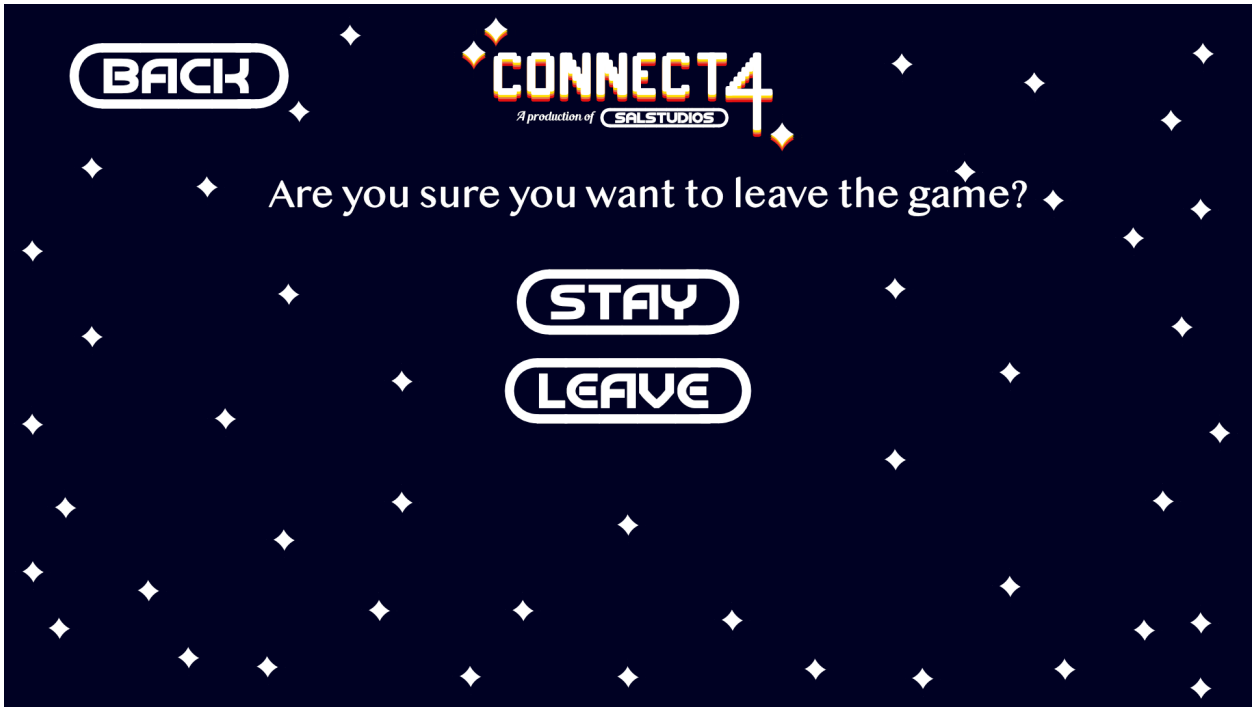
Board Page (Empty):



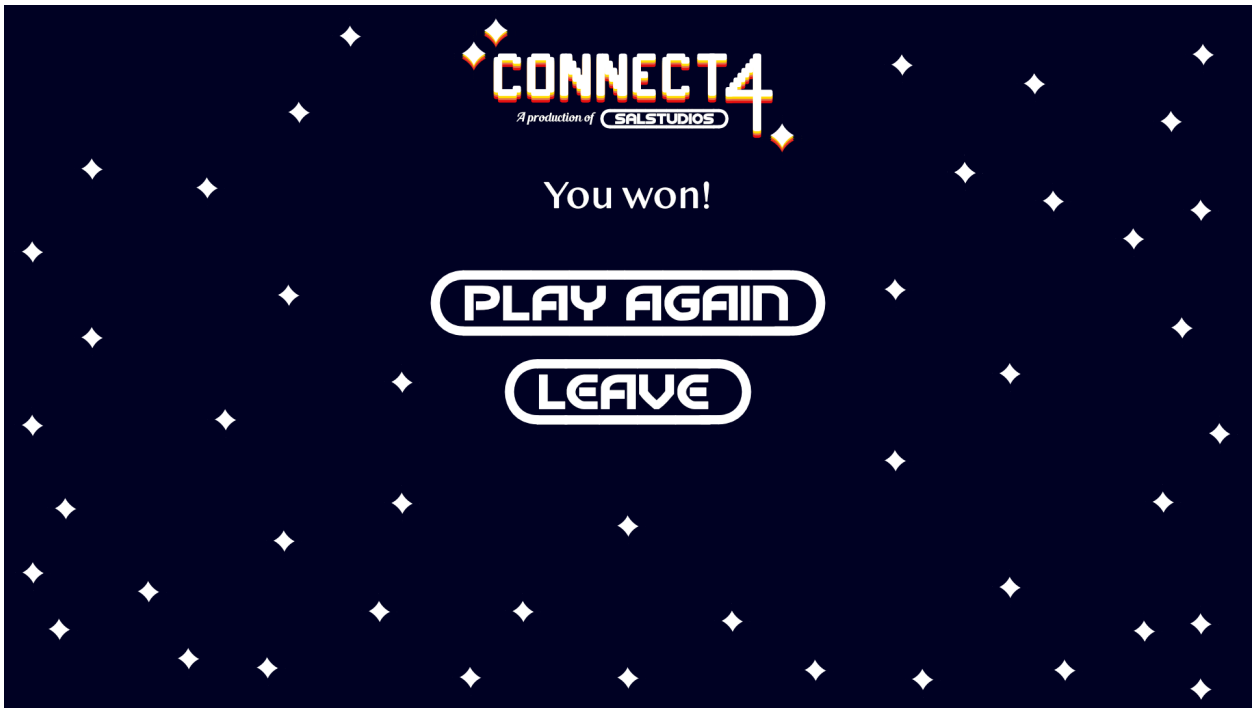
Board Page (Semi-Filled):



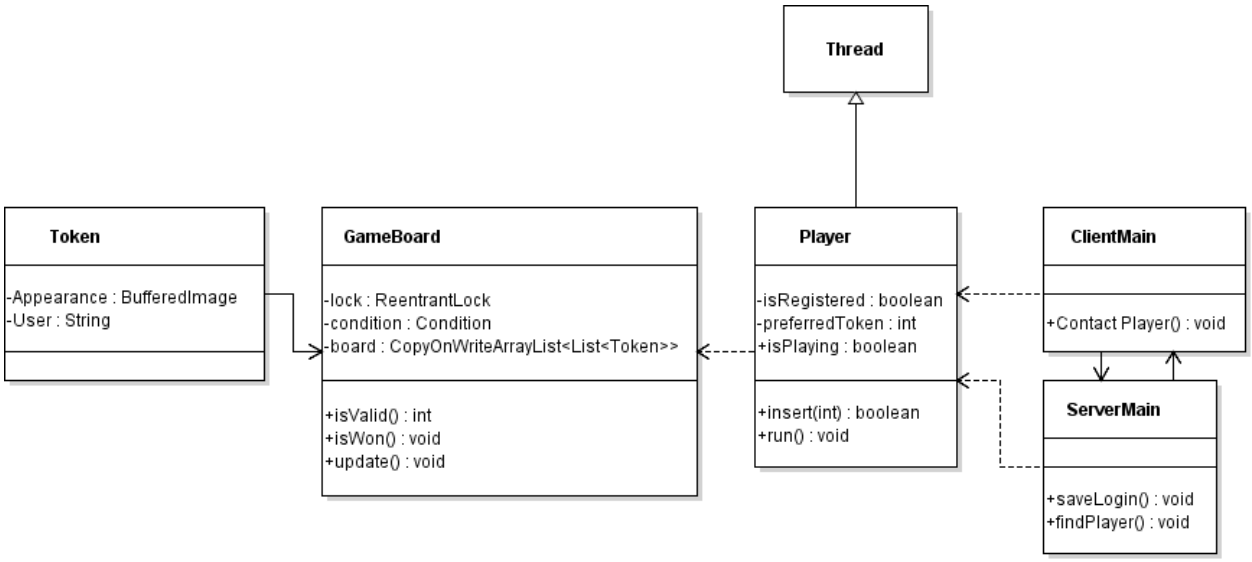
Leave Game Page:



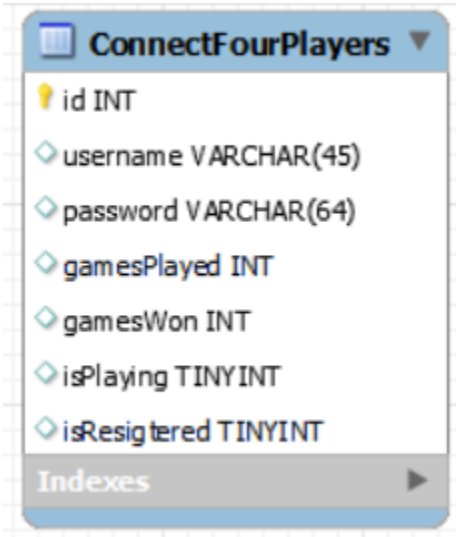
Finished Game Page:



UML Class Diagram:



Database Schema:



Hardware and Software Requirements:

Hardware:

A computer (that's fairly modern)

Software:

Java v14.0.2
IDE: anything that runs Java (Eclipse preferred)