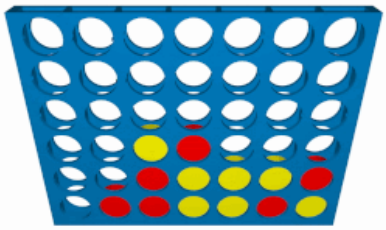
CSC 309 Programming on the Web

Winter 2014

Assignment 3

Connect 4

Due Date: April 4th at 5 PM



Description

This assignment explores the use of AJAX, JSON and database concurrency control.

Develop a web-based version of the classic 70's game Connect 4. The game is played by two players which take turns dropping colored discs from the top into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the next available space within the column. The first player that connects four discs of the same color next to each other vertically, horizontally, or diagonally wins!

To play the game, users first have to login to your site (Figure 1). A user can then wait to be invited to play (Figure 4), or can invite one of the available users (Figure 2 and Figure 3). When a user receives an invitation, they can either accept it or turn it down. If the invitation is accepted the game begins. For simplicity, assume that the user that started the invitation process gets to play first.

Note: All figures are meant to be illustrative. Your application may implement a different (more appealing) UI. The text messaging functionality is provided for testing purposes only; the final application does not have to provide this feature.

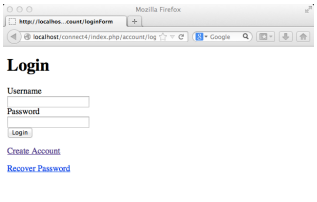


Figure 1

Figure 2

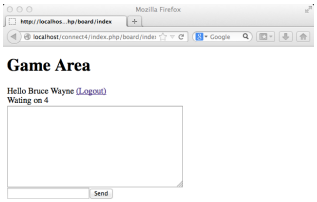


Figure 3

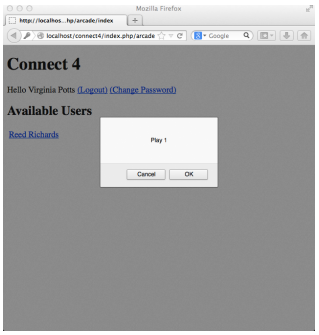


Figure 4

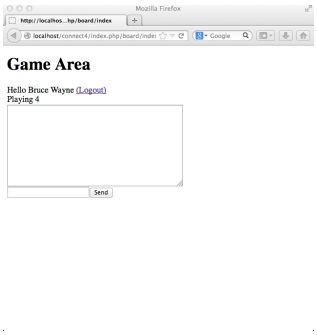


Figure 5

For simplicity, you can make the following assumptions:

- A user can only be involved in 1 match at a time
- Once a match starts, it will continue until it finishes, i.e., users do not disappear or logout midway through
- An invitation will be answered eventually, i.e., no need to timeout invites

Store all your application state on a MySql database that conforms to the schema shown in Figure 6. An instance of this database has been pre-created for you to use. You will get instructions for how to connect to the database in a separate email early next week.

Store the state of the game in the field ***match.board_state***. This field is a blob and will accept an array of bytes. It is up to you to determine how to serialize/deserialize the game state so that it can be store as a blob.

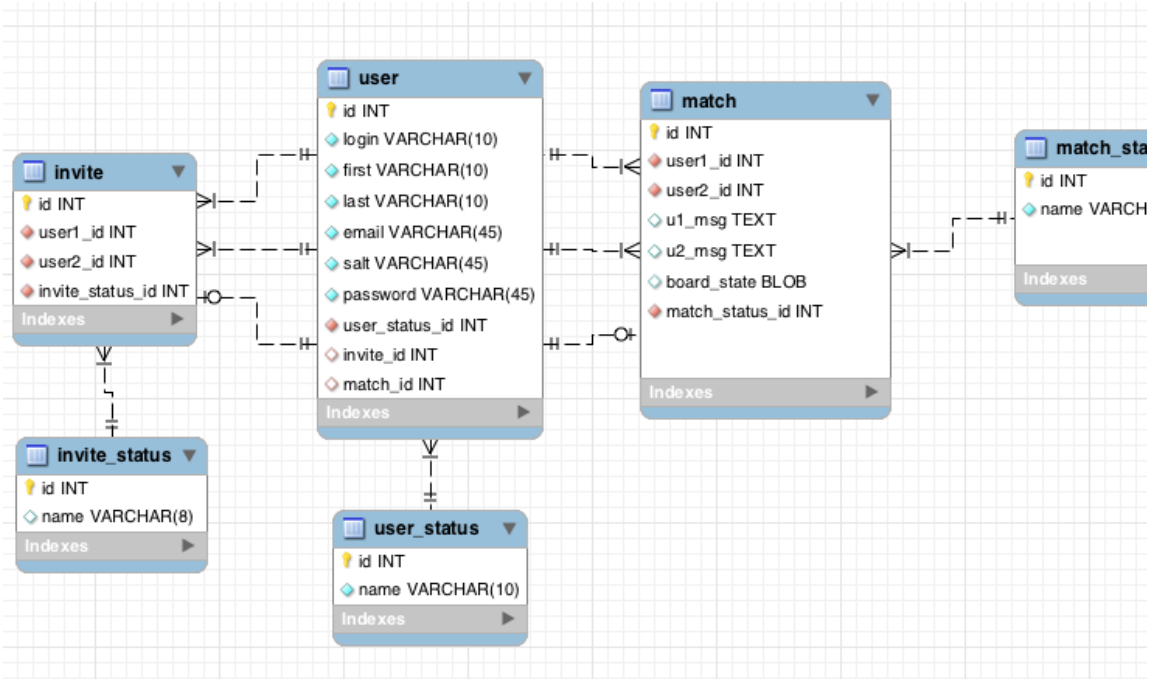


Figure 6: Database Schema

Don't make changes to the database's schema (e.g., add tables, change attribute names). The TA will use their own version of the database to grade your assignment and if you modify the schema, your application won't work. An sql script for recreating the database is available here (you will need this file only if you plan to run MySql on your home machine).

id	name
1	pending
2	accepted
3	rejected

id	name
1	offline
2	available
3	waiting
4	invited
5	playing

Figure 7: user_status Table

Figure 8: invite_status Table

id	name
1	active
2	u1win
3	u2win
4	tie

Figure 9: match_status Table

The user_status, invite_status and match_status tables of the db schema have been pre-loaded with the data shown in Figures 6, 7 and 8. When a user logs into the system, their status is set to ***available***. When a user invites another user to play, the inviter's status changes to ***waiting*** and the invitee's status changes to ***invited***. If the invitee accepts the invitation, the status of both users changes to ***playing***. Otherwise, both users' status changes to ***available***. When a user invites another user to battle, it create an entry in the ***invite*** table with a status of ***pending***. If the invitee accepts the invitation, the status changes to ***accepted*** and to ***rejected***, otherwise. If an invitation to play is accepted, a new entry in the ***match*** table is created with a status of ***active***. The status changes to ***u1won*** or ***u2won*** when one of the users wins the game.

To help you get started, I have created a sample CodeIgniter application called connect4 that provides an implementation of the functionality illustrated in Figures 1 to 5. This application also includes support for text message exchanges between battling users. Texting is provided for illustration purposes; this feature is optional and does not have to be included in your submission.

To use the application follow these steps:

1. Uncompress and untar the file
2. Install it under apache/htdocs
3. Change the values of the database, username, and password settings in connect4/application/config/database.php to the appropriate values for your database.
4. Start Apache
5. Navigate to http://server_name:your_port_number/connect4

To implement this assignment you will need to setup and run Apache. Instructions for running Apache and Eclipse, as well as for debugging PHP are available on the student guide here.

To test your application you will need to use multiple browser sessions. To start multiple firefox sessions on the same computer on CDF type **firefox -no-remote -ProfileManager** and chose a different profile each time. For instructions on how to start multiple firefox session on Mac OS and Windows go here.

Requirements

- Your implementation should make use of the following technologies: CodeIgniter, AJAX, JSON, JQuery, CSS and MySql. You are allowed (but not required) to use additional technologies, such as CSS Animation, JQueryUI, and other JQuery plugins. Use CSS style sheets in favor of deprecated HTML tags and attributes.
- The provided starter code includes support for user authentication based on salted and hashed passwords. Improve the resilience of your site to attack by extending this primitive user authentication functionality by adding a captcha for new account creation. Create the captcha using the Securimage PHP library.
- Make sure your site functions correctly in the CDF environment. TAs will not debug so test to ensure it works the first time!

Deliverables

Submit your assignment using the BlackBoard system. Submit only one copy per group.

Include a README file with documentation for your Web site (brief explanations of how it all works). Make sure the file includes the names and student numbers of both group members.

Submit your assignment as a single file by using a tool like ***tar*** or ***rar*** to archive your complete CodeIgniter directory structure.

Note: Do not submit a copy of your database. The TA will use their own.

Finally, all deliverables should be neatly formatted, readable,and be properly documented .

Good luck!