**Project Deliverable #5: Second Release Report**

1. The customer asked us to implement the **multiplayer** and **chat** user stories. We implemented these stories. We spent about 40 hours total working on the project; compared to our estimate, we were within an hour discrepancy.
2. The Extreme Programming (XP) principles/practices that we followed were almost the same as with the previous release: incremental planning, small releases, refactoring, pair programming, collective ownership, and sustainable pace.
   1. Incremental Planning – this was practiced in the form of the second deliverable our group submitted. We followed the guidelines to list our stories and describe them, including the user tasks and non-functional requirements. Rather than recording the requirements on story cards, we recorded them in a single word document (project deliverable #2).
   2. Refactoring – we cleaned up the layout for all the pages, edited text messages, and made code more readable.
   3. Pair Programming – we worked in pairs; one person coded, while the other checked their work. Then the other person would code and the other would error check. One pair worked on the PHP/JavaScript, the other on the AJAX. Then the pairs swapped assignments. This continued until the second iteration was complete.
   4. Collective Ownership – we made sure that no islands of expertise were developed by doing the steps described in part c.
   5. Sustainable Pace – same thing as with previous release. Our group did not work past 10 PM each day of work. And we did not exceed 8 hours of work per each workday.
3. For the multiplayer user story, we created a game mode called Time Attack. You enter the username of the opponent you wish to challenge. Then you see how quickly you can answer 10 questions and your time is compared to the opponent’s to determine the winner. So the test case for multiplayer was to make sure the game could be played, the time was recorded and displayed and compared to the opponent’s, and then telling the user who won. The inputs are opponent’s username and player’s answer choice. The expected output is “Please enter the username of the opponent you wish to face:”, “Congratulations!!! You've defeated your opponent in this game of wits!” if the player wins, and “Too Bad! Your opponent has bested you!” if the player loses.

The chat test case involved checking if the player could add a friend’s account to their friend’s list and seeing if the player could perform live chatting with one of their friends. To do this, we created a new PHP page for players with an account to manage their friend’s list and chat. The inputs are friend’s username, the message the player wishes to send to their friend, and the message from the friend. The expected outputs are “Please fill in the field” if nothing is in the text field when submitted; “Friend added successfully” if the friend was added; “Account does not exist” if the username entered is not in the database; and the date of when the player messages and friend messages were received, the username tied with each message, and the messages themselves.