**Project Deliverable #6: Third Release Report**

1. The customer asked us to implement the **high score (for single player)** and the **leaderboard** user stories. We implemented these stories. We spent about 30 hours total working on the project; this was very close to the estimated amount.
2. The Extreme Programming (XP) principles/practices that we followed were exactly 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 again, we edited text messages, and we made code more legible.
   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 third iteration was completed.
   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 the first two releases. Our group did not work past 10 PM each day of work. And we did not exceed 8 hours of work per workday.
3. The high score story for single player was implemented by comparing the current score the player has on the current page (so after each question answered) to their existing high score. If the current score is higher than the existing high score, then the existing high score is overridden by the current high score. The test case for this was to make sure that the high score is saved, and then to check that a new high score would override the old high score. We actually checked for this through implementation of the leaderboard user story. The input is the player’s current score on the current page. The expected output is an updated high score if the current score is greater than existing high score.

For the leaderboard user story, we created a new page called “leaderboard.php” that the player could access via the navigation bar. The leaderboard page displays all players and their highest scores, ordering from highest score to lowest score. The test case for leaderboard was to make sure that a player’s high score would accurately display their current high score, and that the board would display the updated high score when a player reached a new high score. Also, when a new account is created, and they answer at least one question correctly, their score should be displayed on the leaderboard. The input is the player’s highest score. The expected output is the player’s username and highest score displayed on the leaderboard table.