Student: 170026484

CS5003

Project 1: Ye Olde Pub Quiz

Due Date: Monday 5th March 2018

Word Count:1,100

Solution Design

The basic implementation was made with this plan:

- **Global variables:** were used to keep tallies of the question number and lives.
- **Fetch URL using a token:** So questions would not be repeated in the same session.
- **Load Questions:** The aim was to have all the information required in separate arrays so they could easily be accessed in future functions.
    - All 10 questions
    - All correct answers
    - All options for every question
        - These were shuffled using a function so that each question options had a different order. This was challenging due to the need to shuffle subarrays.
- **Display Questions:** when the game is started, questions are assigned to the HTML IDs. JavaScript was initially used but after the JQuery lecture, JQuery was used – this made assigning text to the IDs easier. There was an option to cash out or quit (if you have not won any money).
- **User Selection:** this was an event based function that waited for the user's click. Depending on what was clicked, the user would be directed to the following function:
    - **Correct –** add 1 to correct answer's tally. CSS was added to show result
    - **Wrong –** CSS was added to show result. Lose a life: if 3 lives were lost, the game would end.
- **Turn off Options:** I had an initial problem due to the user being able to make many attempts at the same question. Once a user made a selection, this function would prohibit another selection.
- **Prize Fund:** based off the number of correct questions.
- **Next Question:** this would reset the question and add 1 to the question number tally. This would then create a loop that would start at the load question function.
- **End Game:** the game was ended in different ways depending on the method of ending: Quitting, Cashing Out, Answering all questions, and losing all lives.
- **CSS and HTML:** The question was displayed with help from the CS5003 Week 2 problem. JQuery was used extensively to hide and show DIVS and buttons.

Extensions

Once the basic functions were created and working, the following extensions were implemented.

- **Restart without refreshing page**
    - Accomplished by refreshing the global variables and CSS.
- **Timers with 30s lifeline option**
    - Clock: Initial difficulty was getting it to refresh after each question.
    - Visual Count down: This used the same principals as the progress bar taught in lectures but instead focused on using set interval method. I had difficulty synchronising this with the clock especially once the next question button was clicked. *ClearInterval* was used to pause it when the user made a selection.

    - Lifeline of 30s could be added. I liked using the flip-flap of using fade-in/fade-out. This made it stand out. I used this feature for also showing and hiding divs.
- **Change Difficulty and Topic of questions**
    - At the beginning the user could choose a difficulty from a drop down menu. This would be inputted into the fetched URL. The prize fund was adjusted to the difficulty chosen using a switch statement.
- **Bank:**
    - Q4 and Q7 were arbitrarily chosen as check points to guarantee a cash prize.
- **Fifty-Fifty lifeline:**
    - Was used by locating the correct answer and then eliminating two random wrong options.
    - A click counter was used to prevent double clicking it resulting in only displaying the correct answer.
- **Print Leaderboard**
    - An input was used at the beginning to record the name of the user. Their name and winnings were then pushed into an array and then displayed on a leaderboard at the end of the game. The table was sorted to money won.
- **Progress Points**
    - A visual progress bar was used. This used CSS to display the checkpoint (bordered box). It required JS to relate to the question number the user was on (made bold).
- **Ask the audience lifeline:**
    - An idea out of "Who wants to be a Millionaire", I thought it would be good to have a virtual audience. The numbers has to be generated randomly every time
    - In the show, the audience often get the correct answer; but occasionally not. Therefore the weighting scale was used meant the correct answer was favoured to have the most votes.
    - The following considerations were made:
        - The wrong answers votes had to be randomly distributed
            - Using a switch statement dependent on the location of the correct answer the other options were randomly distributed. Long winded but I could think of no other method.
        - All options had to up to 100%
    - Bug:
        - One cannot use 50/50 and then see an audience response of the remaining 2 options.
        - I tried to using many *if* statements but failed to solve it.

Testing

1. Start Screen Example

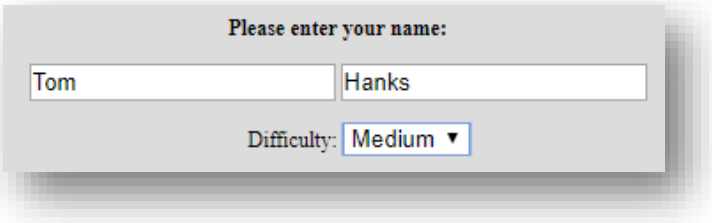

Figure 1: Input example with difficulty selected

2. Difficulty

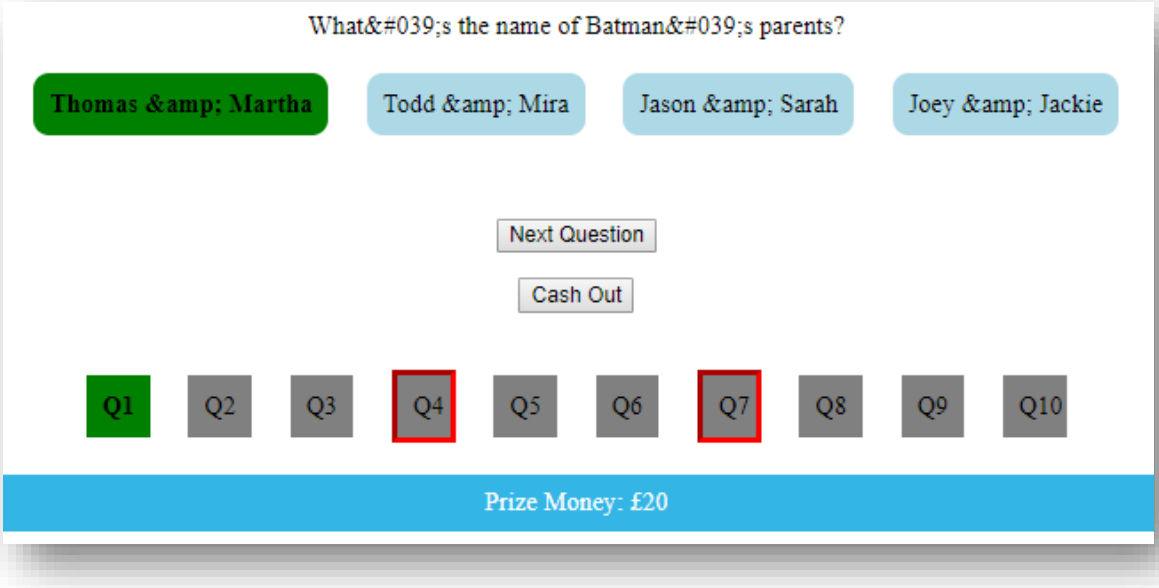

Figure 2: Easy Difficulty = £20

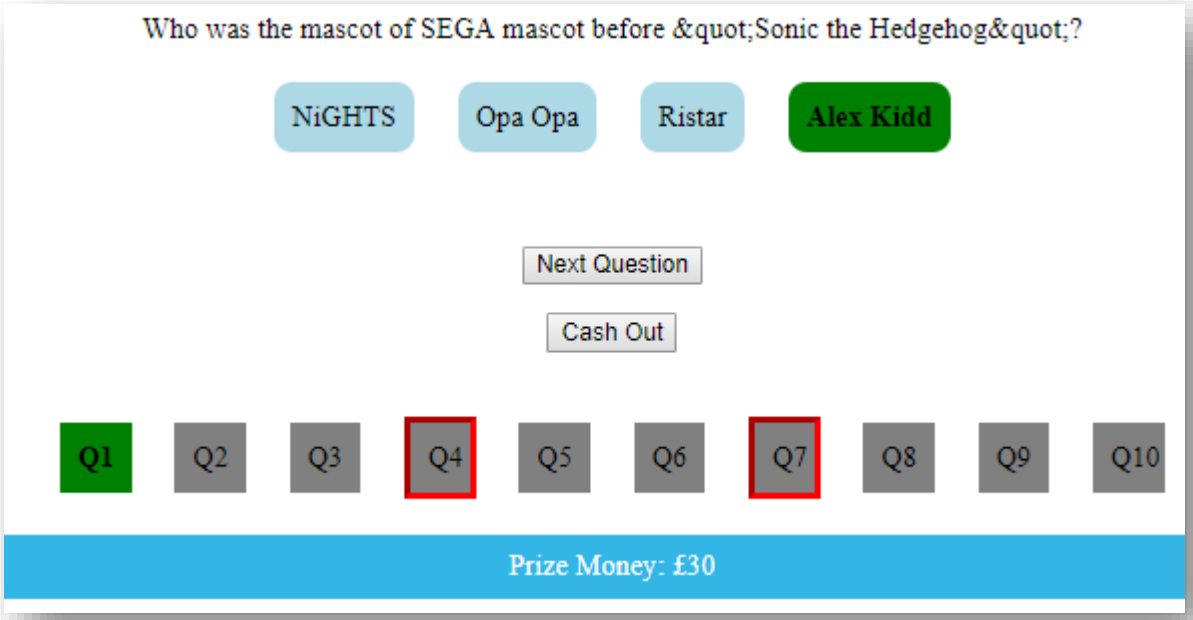

Figure 3: Medium Difficulty = £30

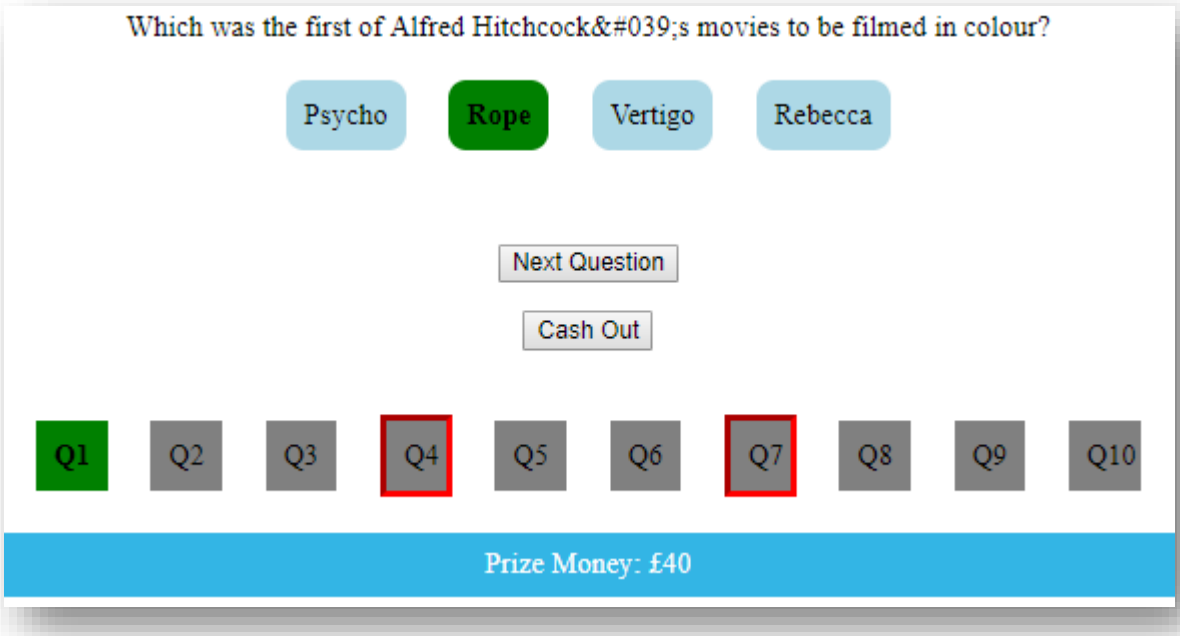

Figure 4: Hard Difficulty = £40

3. User selection

3.1 Wrong Answer

- Cannot click on another option
- Quit offered instead of cash out
- Life lost
- Timer stops

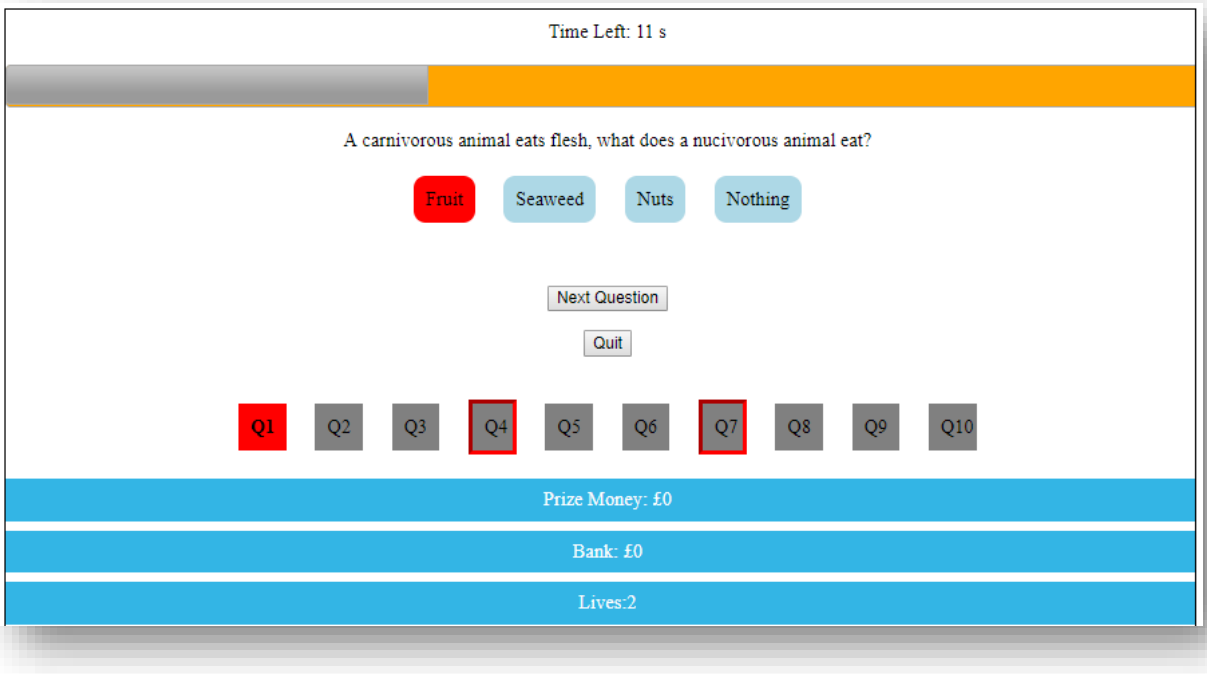

Figure 5: Wrong answer selected

3.2 Time Out

- Can't select any option
- All options are red

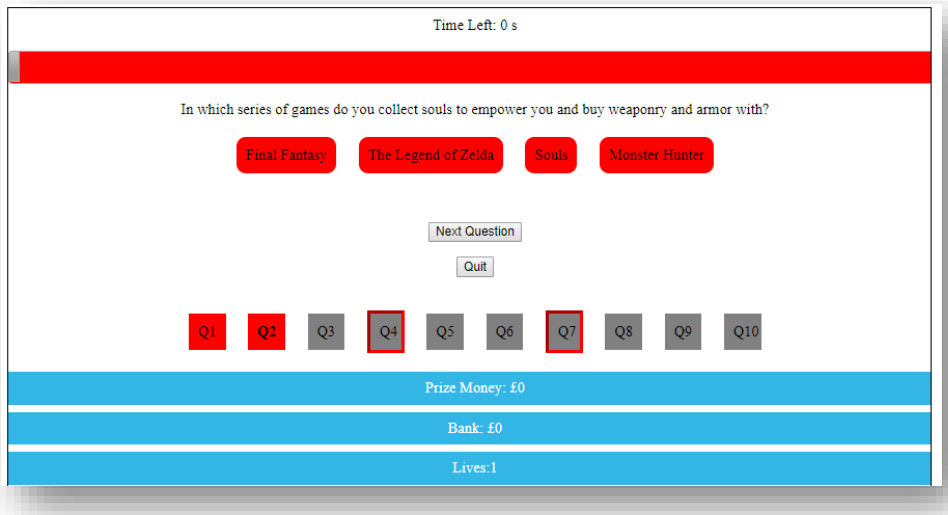

Figure 6: Time run out

3. 3 Add 30s

- Add 30s button is removed
- Button disappears
- Time is 55s

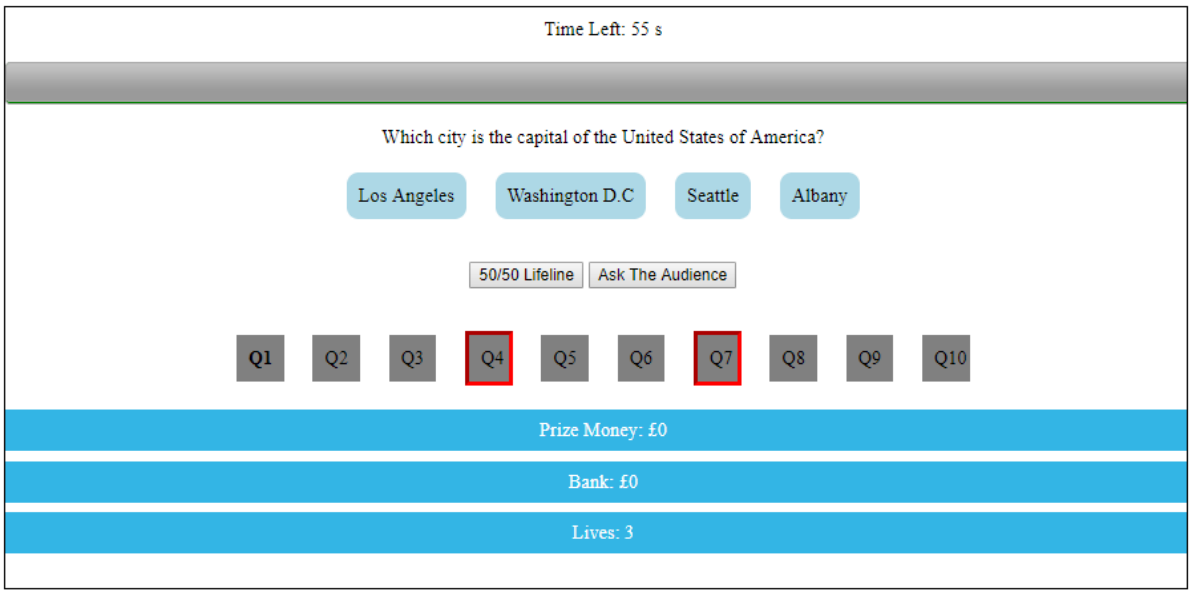

Figure 7: 30s added

3.4 50/50

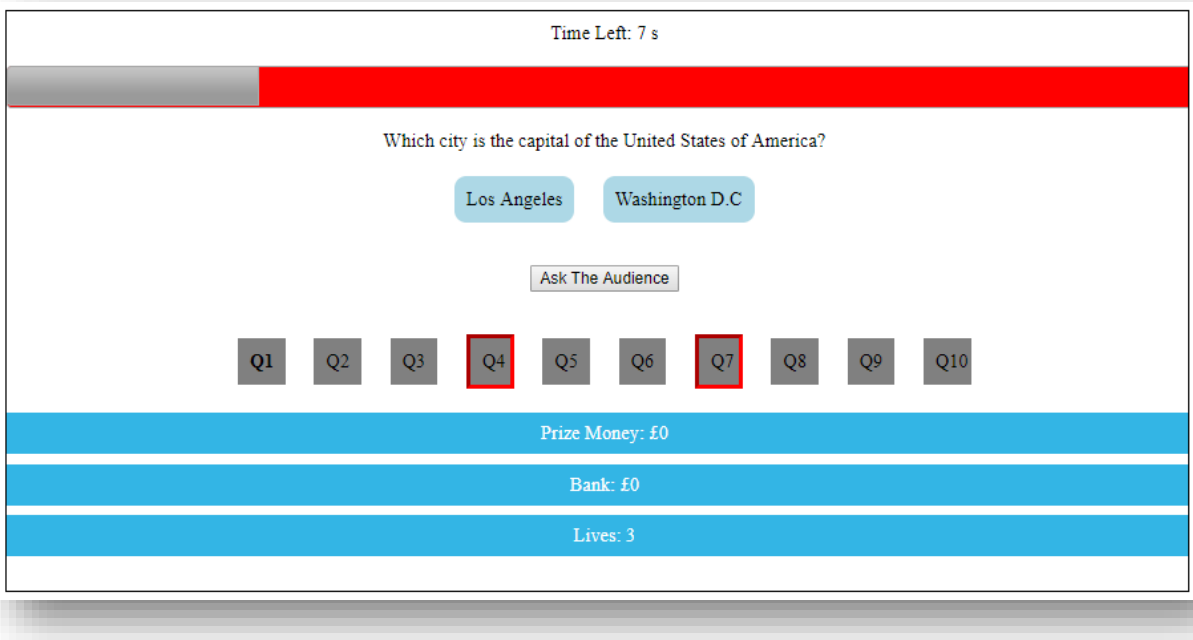

Figure 8: 50/50 button selected

3. 5 Ask the audience

- Sum of percentages is 100
- Button disappears
- Psychoanalysis is the correct answer and was rightly displayed as favourite amongst the “audience”.

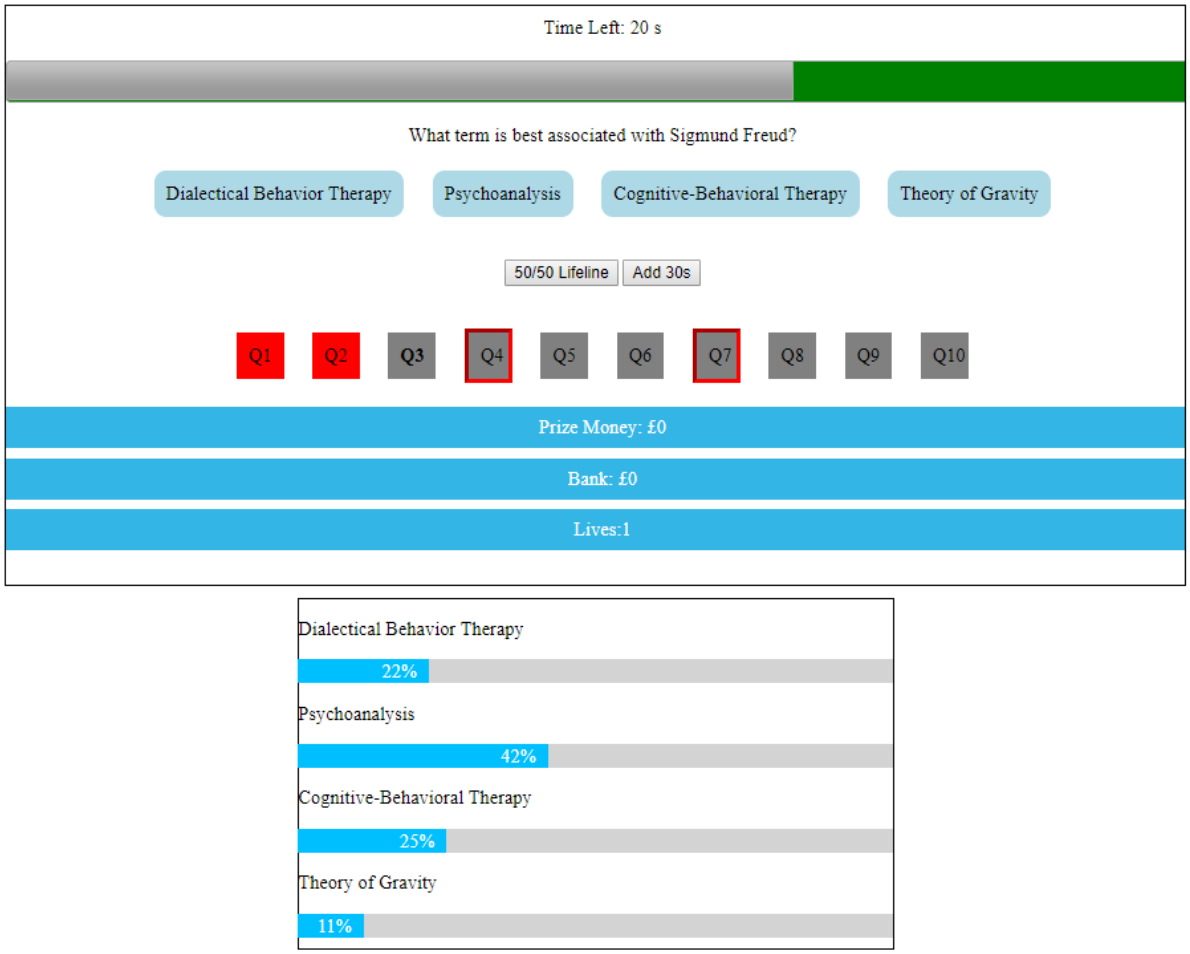

Figure 9: Ask the audience button selected

In this Figure 10 the wrong answer is actually favourite. The correct answer is 40 – as seen in the console.

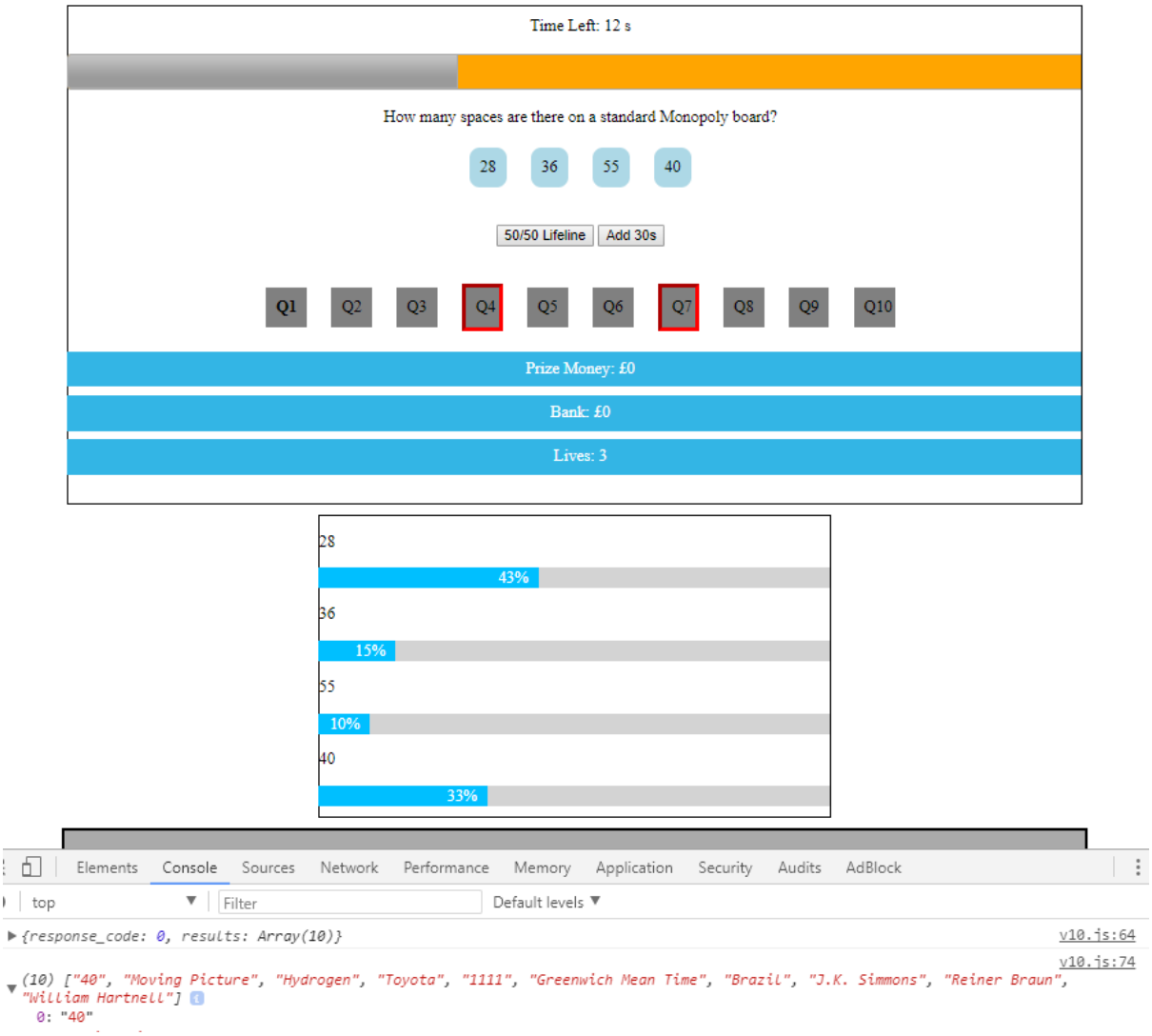

Figure 10: Audience guess wrong

4. Checkpoints

- Checkpoint 4 is bordered green to show it has been reached.
- Checkpoint 7 is bordered red to show it has not been reached.
- £90 has been banked even though the person has £120 cash.
- The user can cash out if they want

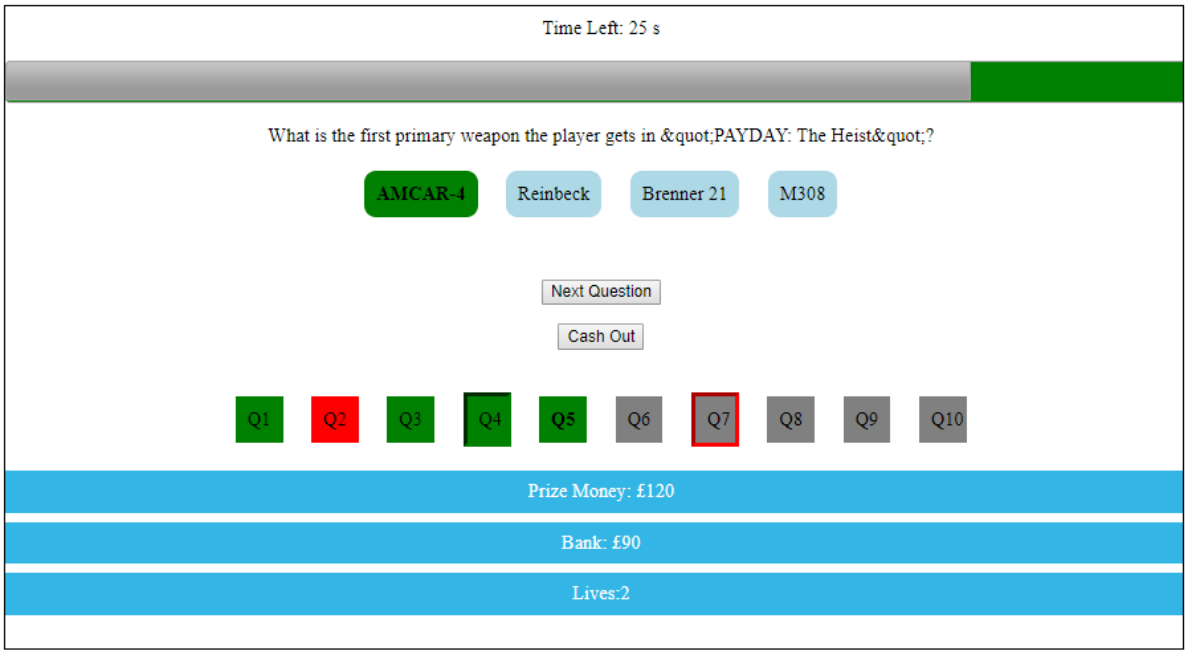

Figure 11: Going past a checkpoint

5. End of Game

5.1 Completed the game

- Score is recorded in leaderboard

Figure 12: Completing the game

5.2 Loss of all lives

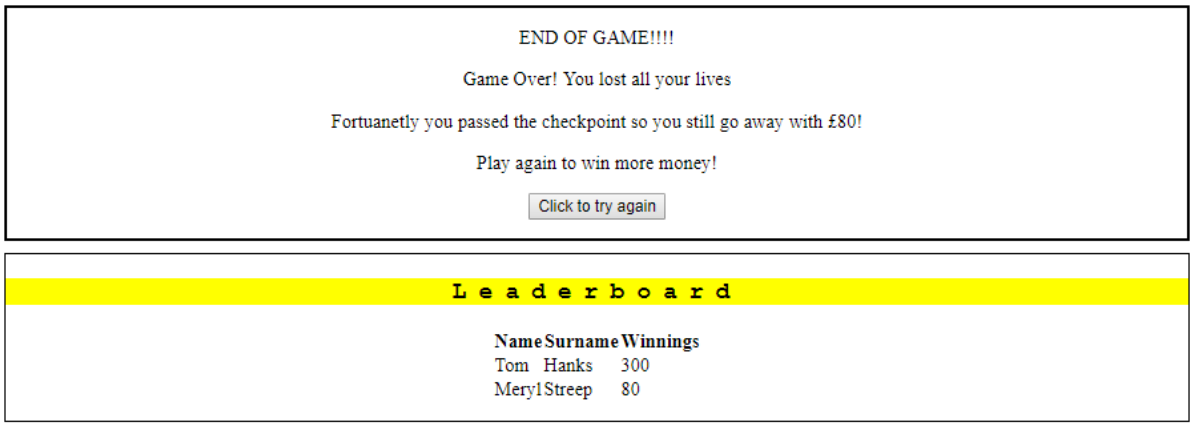

Figure 13: Losing all lives

5.3 Bank

- Had £120 prize money and £100 in the bank
- Lost all lives but came away with £100

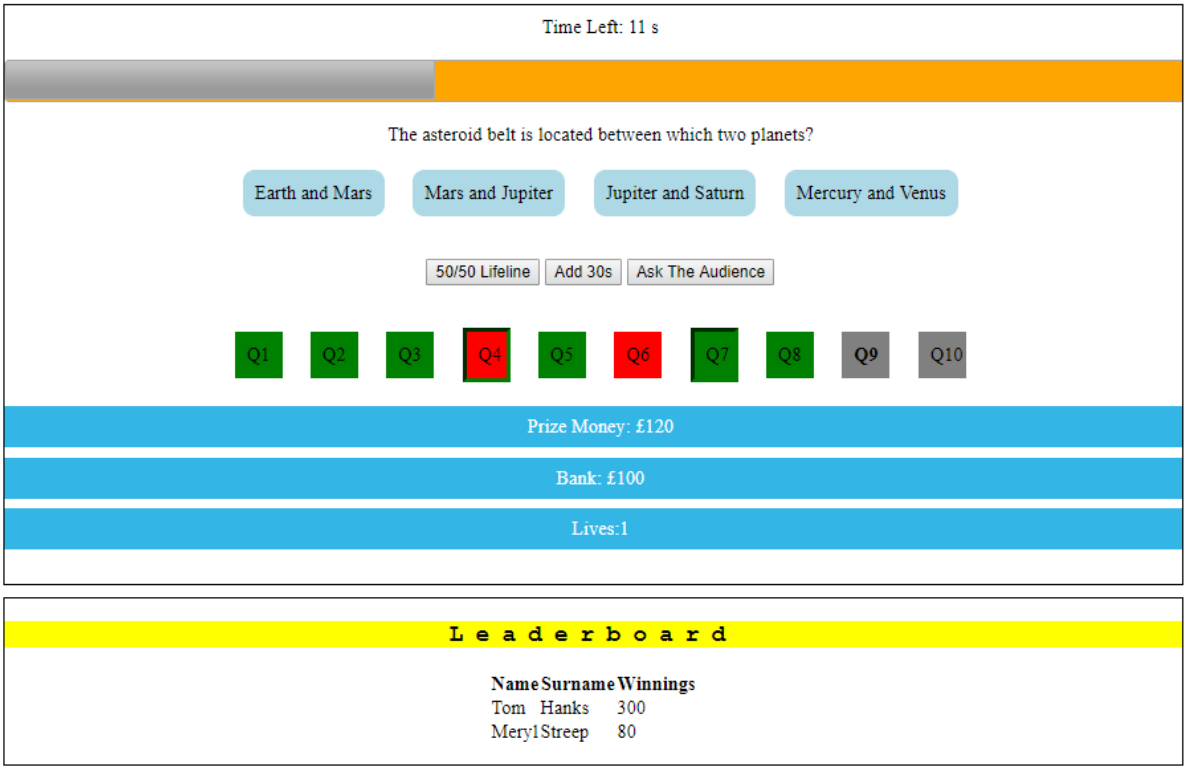

Figure 14: £100 in the bank; £120 is prize money

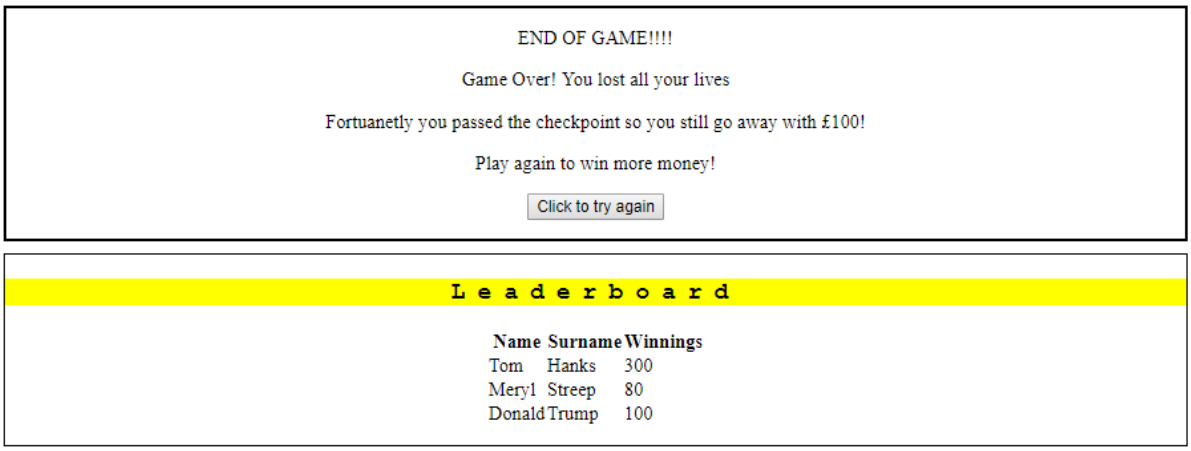

Figure 15: Losing with money in the bank

5.4 Cash Out

- Entry not recorded in table

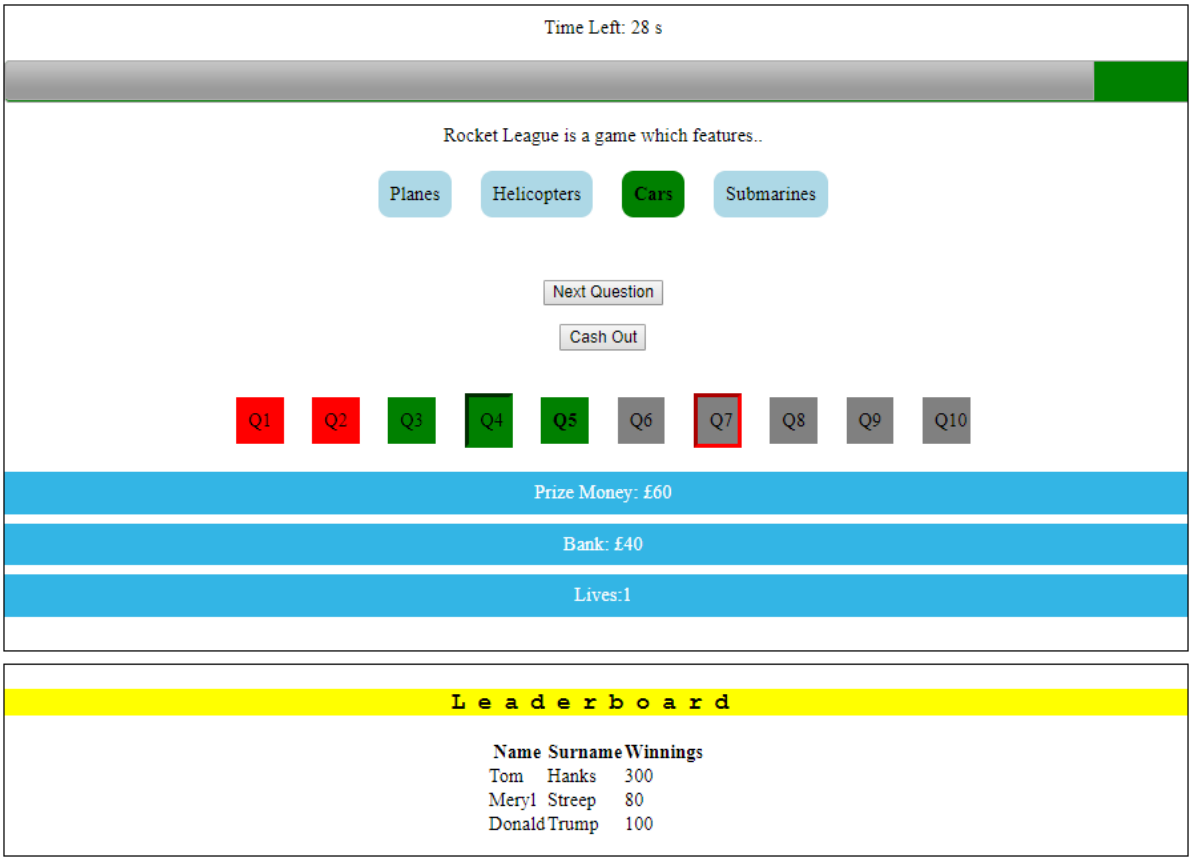

Figure 16: Selecting the cash out button at this stage

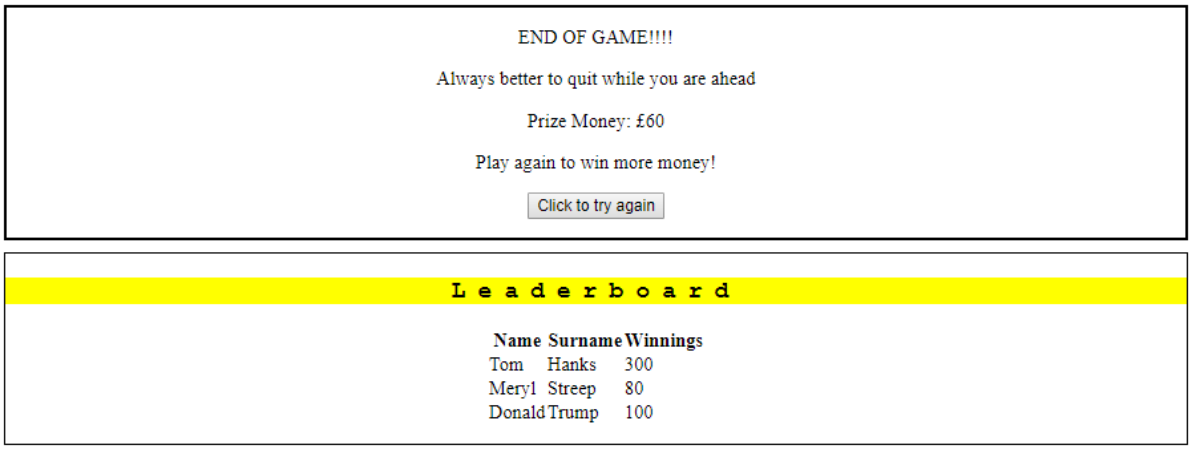

Figure 17: Clicking "cash out" in Figure 15 results in this screen

5.5 Quit

- Left with £0

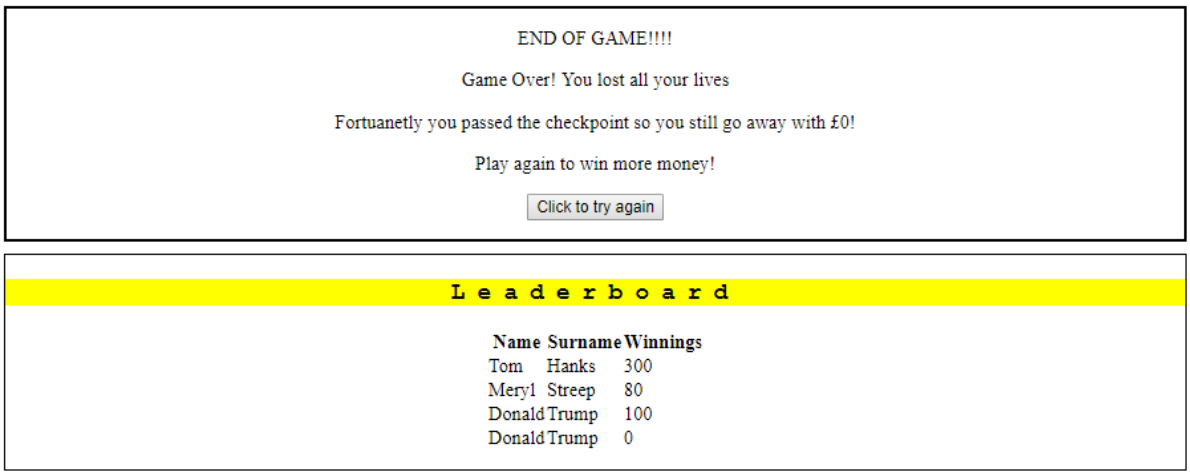

Figure 18: If user quits having not made any money