Coursework Report

Bruce Morel-Barnes
40202779@live.napier.ac.uk
Edinburgh Napier University - Advanced Web Technologies (SET09103)

1 Introduction

For the second coursework in this module I had initially decided on making a multiplayer browser based game, eventually this centered into a card game because I am familiar with the rules of blackjack. The system in place allows for any person to make an account (which is stored in a database) and begin playing blackjack with another person. This is through a sort of queuing system I devised, where the server searches for 30 seconds until another player does the same or time runs out. From there data is shared between the two users via the server, then allowing the players to play against each other.

2 Design

The aesthetic design is very simplistic and makes use of a folder of .png files for each playing card to give a graphical representation of what is happening in the game. My initial idea had been to have a screen for the log in page, which serves self explanatory purposes, and the sign up page, which I swiftly realised could be rolled into the log in page (albeit I have to imagine that it is less secure than having multiple levels to the system). I had then made designs to implement separate player profile and game finder pages, this idea was scrapped and they were merged into one for ease of use and to simplify the interface. Then, of course, came the game screen whose designs were far more complex than what has been handed in, originally I had wanted to implement a dealer role and have a relative perspective of cards, where players would not be able to see each other players' cards. I had to settle for a simpler approach where each player can see all cards in play and any player ends the game by clicking the "stick" button (this should normally simply pass to another player, after that player has taken as many additional cards as required).

I decided to use bootstrap, this aided greatly for placing objects at specific locations on the web page. I was going to use flask-socketio to make it so that the browser didn't need to be refreshed to show the changes, unfortunately I was not able to identify how to use it within the time allocated. Instead, I had to use a meta tag in my jinja2 template to force each instance of the blackjack game to refresh every 10 seconds, which turned out to be very inefficient since it reloads each image file with every refresh.

3 Enhancements

There are several improvements that could be made, several of which I have touched on previously. The lack of an efficient refreshing system, which brings the overall smoothness of the play experience as it makes loading the current gamestate take a second or two and means that it is only loaded once every 10 seconds, which makes the game very slow and unintuitive. Along the same lines: there are several shortcomings in the rules that have been implemented, for example the ability to stop getting cards by "sticking" is poorly done as the way in which it has been means that the game ends and each player's score is totalled and compared to one another to ascertain a winner. One more issue with that is that when one player clicks the "stick" button, the other player is left on the game screen with no feedback whatsoever.

Another part that needs improvement would be the lacklustre aesthetic design, which looks incredibly bare bones and it is not very pleasant to look at. This is amplified by the fact that the blackjack rules are not properly implemented, these issues could be solved by adding some more content to each template and implementing the game rules more effectively.

4 Critical Evaluation

I think that of all the features in the program the implementation of the database with sqlite3 was the most successful portion as it works perfectly according to how I envisioned it working. As previously mentioned I think the only issue with it would be the potential lack of security with how data is passed around, the main issue I could see with that, however is (assuming that it was not possible to see all of the cards no matter who you are) that someone could cheat by looking in the html to find that the image names are identical to what they represent i.e. 5S.png is the 5 of spades. This would pose a problem as it would allow players to boost their own score and win without being good at the game. I do believe that not storing the password anywhere other than initially comparing it to the database is helpful as I expect if it had, it would be significantly easier to fish out users' details than it is currently.

On the other hand, there are many parts that have already been mentioned that were wanting, but other than those mentioned, I think that the way in which I reused several large chunks of code over and over, in hindsight I should have made a function for each of those sections, it certainly would have made the code more readable if nothing else.
Once again, my implementation of the basic rules of the game were bad at best, since the core elements are missing e.g. you can't get "blackjack" (any ace and any face card) to instantly win, players do not "bust" at over 21 points, essentially meaning that it's up to the players of the game to not cheat each other out of wins and accept a loss when they should. Another function that is missing is the ability for a player to "fold" i.e. to give up and concede the current game to the opposing player. The one redeeming factor is that my function for each player drawing their hand works consistently and should not break easily (unless all cards are drawn from the deck where an infinite loop will occur, this is impossible in a normal game of blackjack and thus is a non issue).

5 Personal Evaluation

Alongside several other projects this semester, I have found myself designing the program before implementing it far more frequently, this has been unbelievably useful with complex problems like this one, I would say that I have grown to appreciate doing so far more than I ever have before. I found it fascinating how easy sqlite3 was to implement and use in and out of the project, and think that for future projects of similar styles I would find myself using it again mainly because of its simplicity and how useful databases often are.

The greatest challenge I faced was finding a way to link players 1 and 2 together into a single "lobby" and loading in to what was more or less the same game (albeit there were the refreshing issues and the various other problems listed above). I do feel that with more time invested into this project, learning how to use AJAX to more efficiently keep the web page refreshed, it could become a far more functional game system with multiple games instead of there only being a barely working blackjack game.

The linking was by far the most interesting part of this project for me mainly because of how challenging the experience was, I hope that I'll be able to put this to use in another project in the future.

6 Conclusion

Overall, this game failed in it's attempt to be a multiplayer game of blackjack (mostly due to the fact that the rules of the game were not implemented to a satisfactory degree), however it did succeed in linking two players' data to make a baseline for future iterations or variations of a game that is functional. I deem this to be more important than the game itself, as any game could have been used to demonstrate the efficacy of my game queue algorithm since it was never actually linked to the game, and just the data (which is completely replaceable).
A potential variation on the algorithm could be something non game related for example a multiple user recipe program which would be several people doing the same thing, potentially with a chat option to give each other advice.

7 References

In this project sqlite3 was used to create and modify a database for the user accounts of individuals who would be players (storing their Username, Password, number of Wins, and nubmer of Losses).

The card .png files were sourced from: http://byronknoll.blogspot.co.uk/2011/03/vector-playing-cards.html