**CSCE 4523 Database Management Systems**
**Project**
**Due: Wednesday, April 21 at 11:59pm**
**By: Joshua Davis**

## Objectives

The goal of this assignment is to apply our previous knowledge of databases to a web front with free range for selecting a coding language and theme. The project must have at least 3 tables with support for at least 6 functions.

## Approach

### Theme/Tables

Regarding the theme, as an avid collector/player of Magic the Gathering, I was excited to implement a database of sorts for sorting cards. I first modeled out my table structure and decided on having a table for Players, Cards, Collections, and Decks. Players have an id number as their primary key, name, and a favorite color (in relation to the game).
The data for cards was thorough, as I was able to extract the information and put it into an excel sheet that used cell functions to properly format the data on a new page, ready for pasting into MySQL. Since magic has a massive library of unique cards (over 20000), I opted to only use a single set for the data (290 cards). I did leave room for adding additional sets with my code though, as the primary key for magic cards need only be {set, set_number} since card names are not unique among sets.

The Collections table keeps track of each player's card inventory that are not currently in decks. It contains columns for the owner_id, card_name, card_set, quantity, and - "in_use", for those that are in decks. The deck table is similar, containing data for owner_id, deck_name, card_name, ***card_set***, and quantity.

### Website/Code

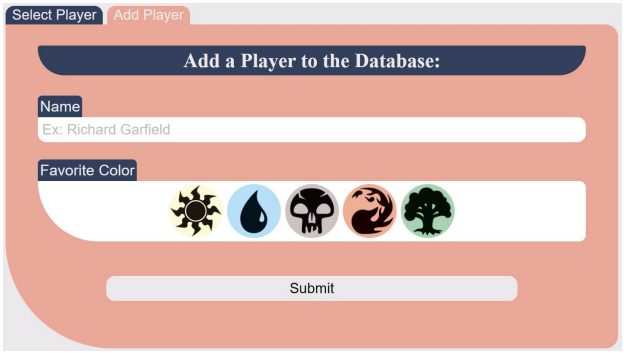
From here I started building up my website. I started with the C++ sample code and put my project on Turing, accessible with the link:

http://www.csce.uark.edu/~jad041/project_cpp/odbc_main.php

I had a specific idea in mind; I heard we were supposed to make it navigable without using the back button at all, but I took it literally and just made everything show up in one php file. As I continued working with the C++ starter code, I noticed that there were a few things I could not get it to do, no matter how long I tried. I ended up finding the php example website posted along with the project and started using it as a hybrid database system for interfacing with elements through jQuery. That allowed me to update my page elements immediately without the need to leave the page.
I still left in the code I had for the C++ functions and ended up adding more later.

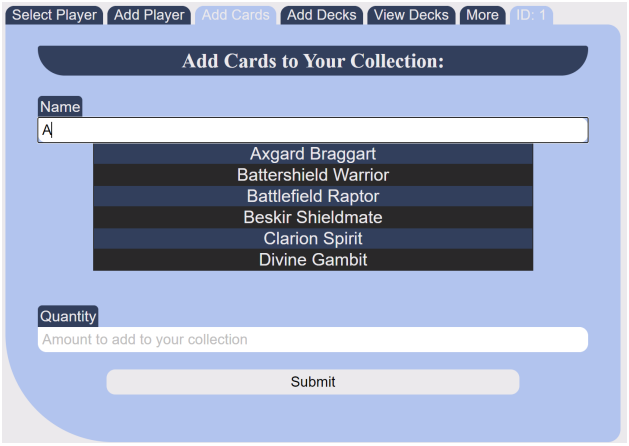
## Results

The major functions I have implemented are:
[1] – Select a Player:
Display an interactive list of all players and their respective ID's. When clicked, effectively 'log in' to that player so that they can make changes specific to them. Besides "Add Player", the other tabs remain hidden until you choose an ID.


[2] – Add a Player:
Enter a name and click on a mana color in the menu to create a new player. User is redirected to the 'Select Player' tab, where the new name is now listed. (this is a side effect of my C++ code, most of them all must return to this tab to update).

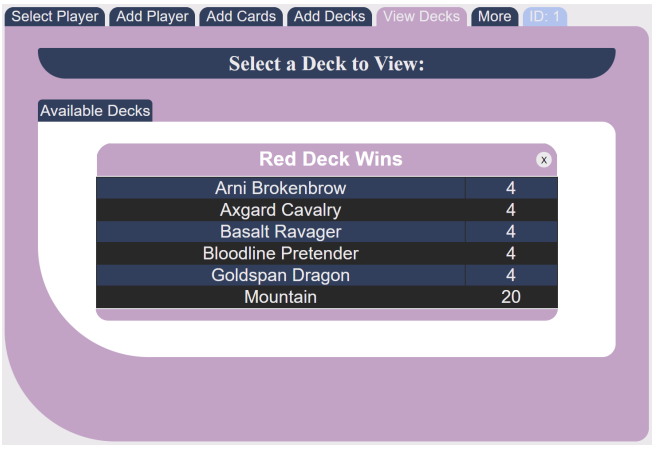
[3] – Add Cards:
Enter the name of a card and the quantity you wish to add to your collection. Under the name box, a jQuery function runs and suggests cards from the CARD table. Entering a card that is not in the database this way produces a corrective error.





[4] – Add Decks:
Input a deck name and select cards from a list populated based on the active user ID and their cards from the COLLECTION table. Change quantities on the right and hit submit to add the whole deck to the DECK table and make those cards unavailable in the COLLECTION table. Error checking disallows inputting more than the quantity owned or less than 0 (it just uses HTML).

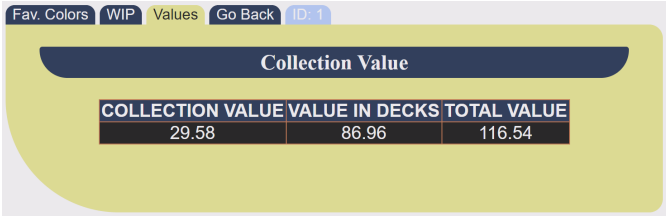
[5] – View *& Delete* Decks:
Any decks that have been made will appear here, separated into their own lists. Each list has an (X) button where the user can delete the deck. When pressed, an alert pops up to confirm the decision.





[6] – Favorite Colors
Divide up the users based on their choice of favorite color. This one is quite simple, but it looks kind of neat when you get a bunch of names on there. *Apparently I like this format or something, cause I did it with these screenshots too.*

[7] – Get Value *(Bonus?)*
Check the total value of every card in your collection, every card in your current decks, and the combined total of the two. The value for each card is found in the CARD table with the rest of the detailed card info I extracted.



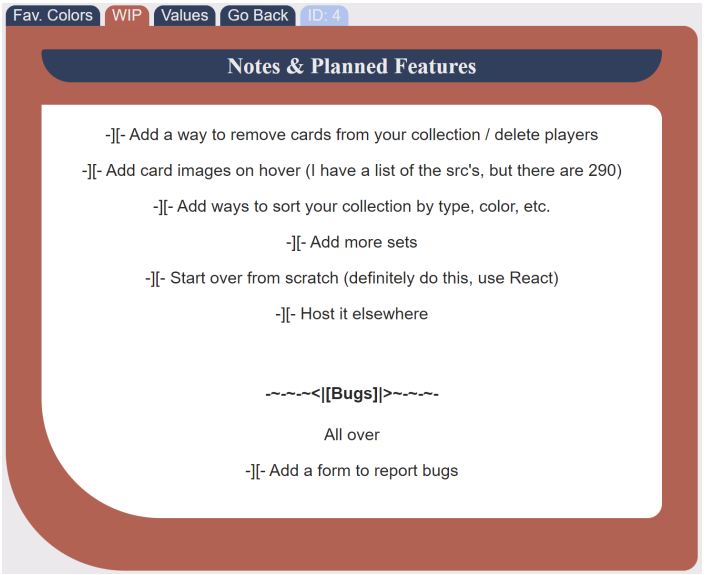

Overall, I am happy with how the project has come along. As far as before and after tables go, most of these are the results of inputs from one another, so I am not entirely sure what to include in that regard screenshot-wise. Regardless, I added a before and after screenshot at the end of this report.

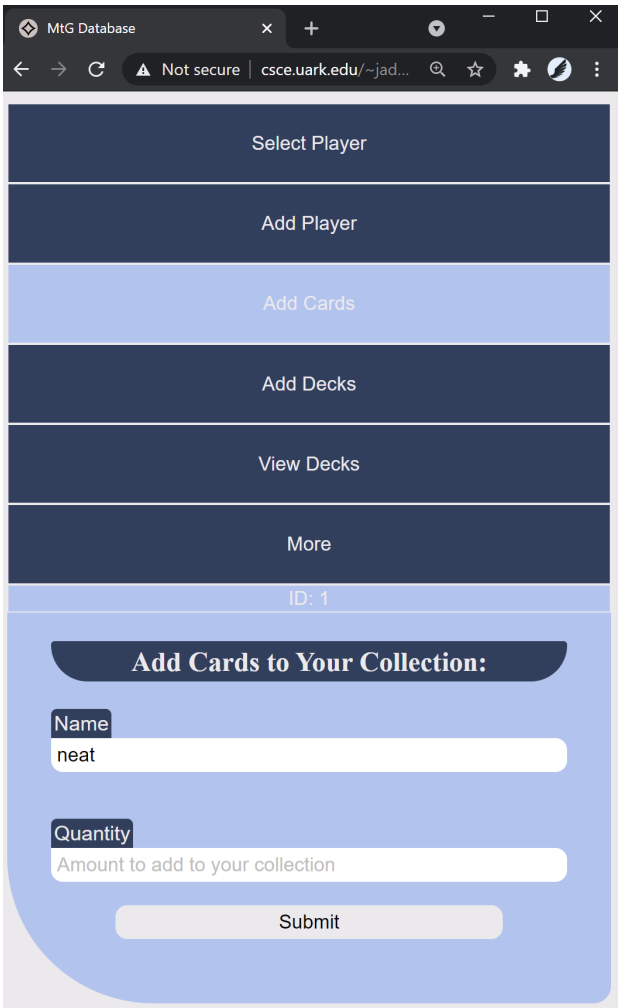
## Testing

I typed everything up in Visual Studio and figured out how to connect it to Turing, which has been very convenient. To get formatting right I would edit divs in inspection mode on browsers for quick changes. There were plenty of headaches and problems trying to get C++ to do what I wanted (like conveniently add cards without logging you out). There were also struggles trying to get php to send variables over to new documents, and just getting a general understanding of how that whole language likes to work. To keep track of the bugs and whatnot, I added comments thoroughly throughout the document, made notes like below, and just handled them as I came across them. I also tested the code on my phone and realized it was pretty horrendous looking, so I added a phone layout mode, as shown below. In hindsight, I would most certainly start over if I were to make this again, as the mix of php and C++ queries can get a bit odd.

## Conclusion

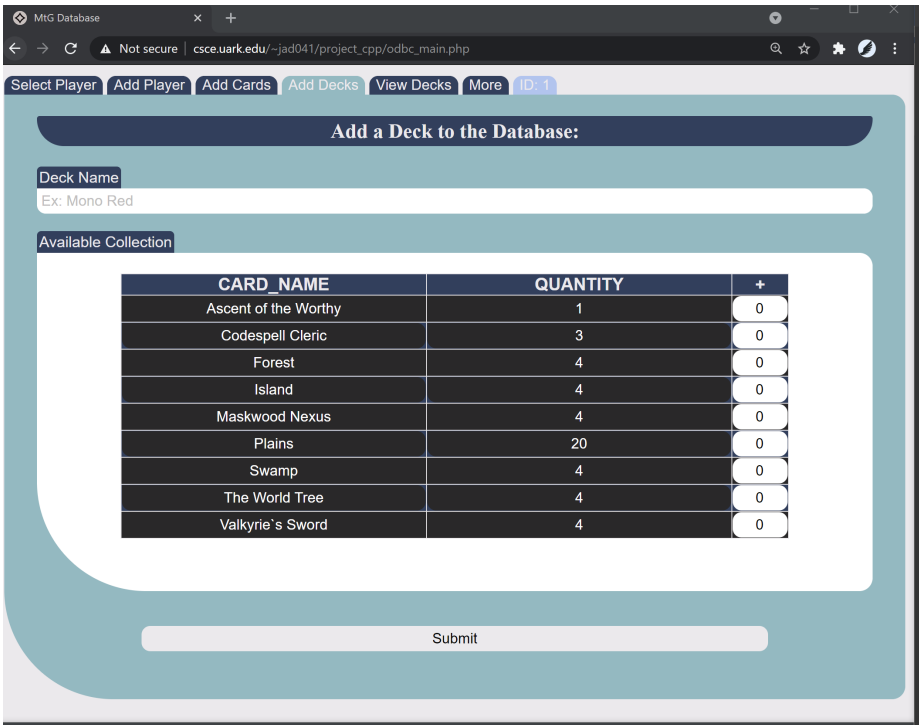
I believe the project was a success, and I got a chance to work with some code I did not have too much experience with. I had a whole lot of fun working on this project and I will be adding more to it. I felt like I was able to learn quite a lot throughout this project and I hope to put it to good use in the future.

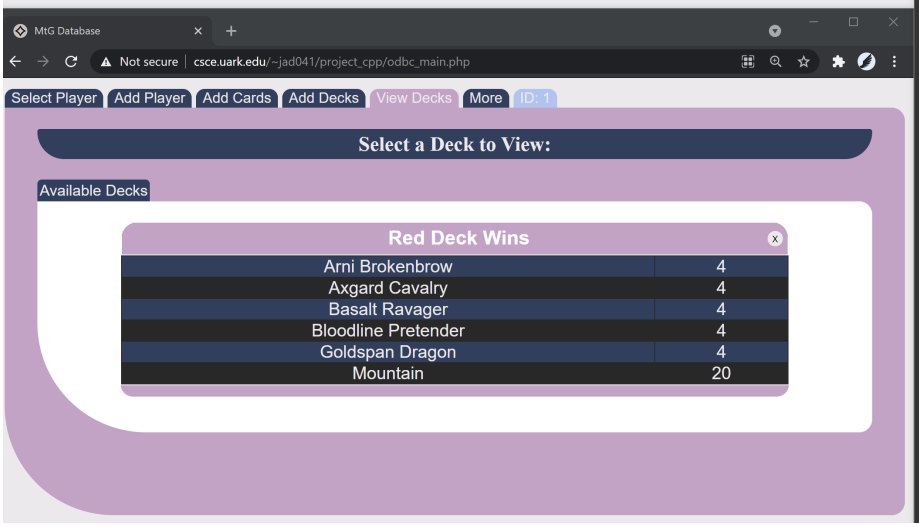


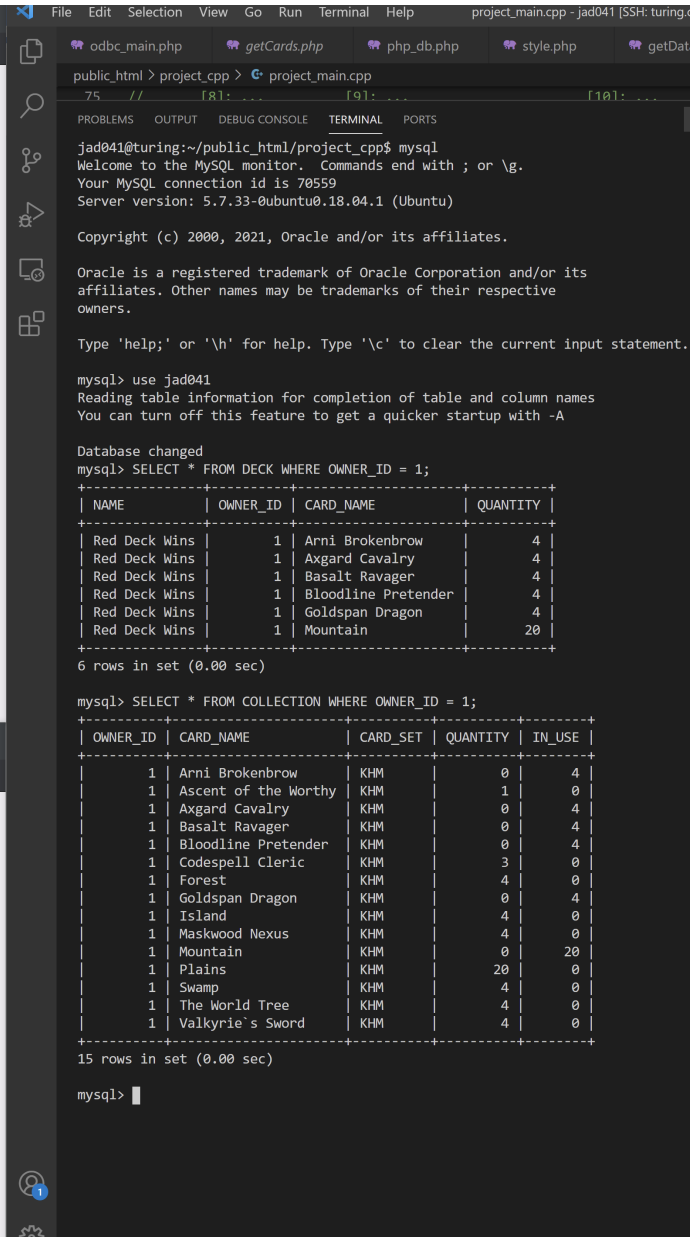



## Before and After Screenshots

### COLLECTION and DECK before deleting a deck



## After deleting deck