**CSCE 4523 Database Management Systems**

**Website Project:** [**http://csce.uark.edu/~krs028/index.php**](http://csce.uark.edu/~krs028/index.php)

**Kyle Sadler**

**Objectives**

The objective of this homework is to create a web application that serves as the frontend to a relational database. The web application should allow the user to perform six functions on the database (and a seventh for bonus), such as viewing, creating, and updating the database’s records and performing complex searches on the data. The database may contain any type of data but should include at least three interrelated tables. The web application may be implemented using any programming language or combination of languages.

**Approach**

I chose to host my web application on http://turing.csce.uark.edu using its built-in Apache web server. My application was written in a combination of PHP and Python and used MySQL as its relational database management system. PHP was used to generate static HTML web pages and call backend functions, while Python was used to interact directly with the MySQL database. JSON was used in order to efficiently pass objects between the PHP and Python backends.

I chose to stick with the basketball tournament theme suggested by Dr. Gauch. However, I modified the tables and attributes to better suit my vision for this project. My database consisted of four tables: TEAM, PLAYER, GAME, and RESULT. The table designs are shown below.

|  |  |
| --- | --- |
| **TEAM** | |
| ID | Integer |
| NAME | Varchar |
| MASCOT | Varchar |
| TOURNAMENT\_SEED | Integer |

|  |  |
| --- | --- |
| **PLAYER** | |
| ID | Integer |
| NAME | Varchar |
| POSITION | Varchar |
| TEAM\_ID | Foreign Key (TEAM.ID) |

|  |  |
| --- | --- |
| **GAME** | |
| ID | Integer |
| COURT\_NUMBER | Integer |
| HOME\_TEAM | Foreign Key (TEAM.ID) |
| AWAY\_TEAM | Foreign Key (TEAM.ID) |
| DATE | Date |

|  |  |
| --- | --- |
| **RESULT** | |
| GAME\_ID | Foreign Key (GAME.ID) |
| HOME\_SCORE | Integer |
| AWAY\_SCORE | Integer |

Each table’s ID field served as the table’s primary key, except for the RESULT table which used the foreign key GAME\_ID as its primary key. All attributes were required to be not NULL, and all foreign keys were restricted on delete. The database was seeded using four teams, Arkansas Razorbacks, Kentucky Wildcats, Harlem Globetrotters, and LSU Tigers, as well as sixteen players, six games, and three results.

The web application consisted of one home page and seven function pages. Each function page performed one or more database functions as described below.

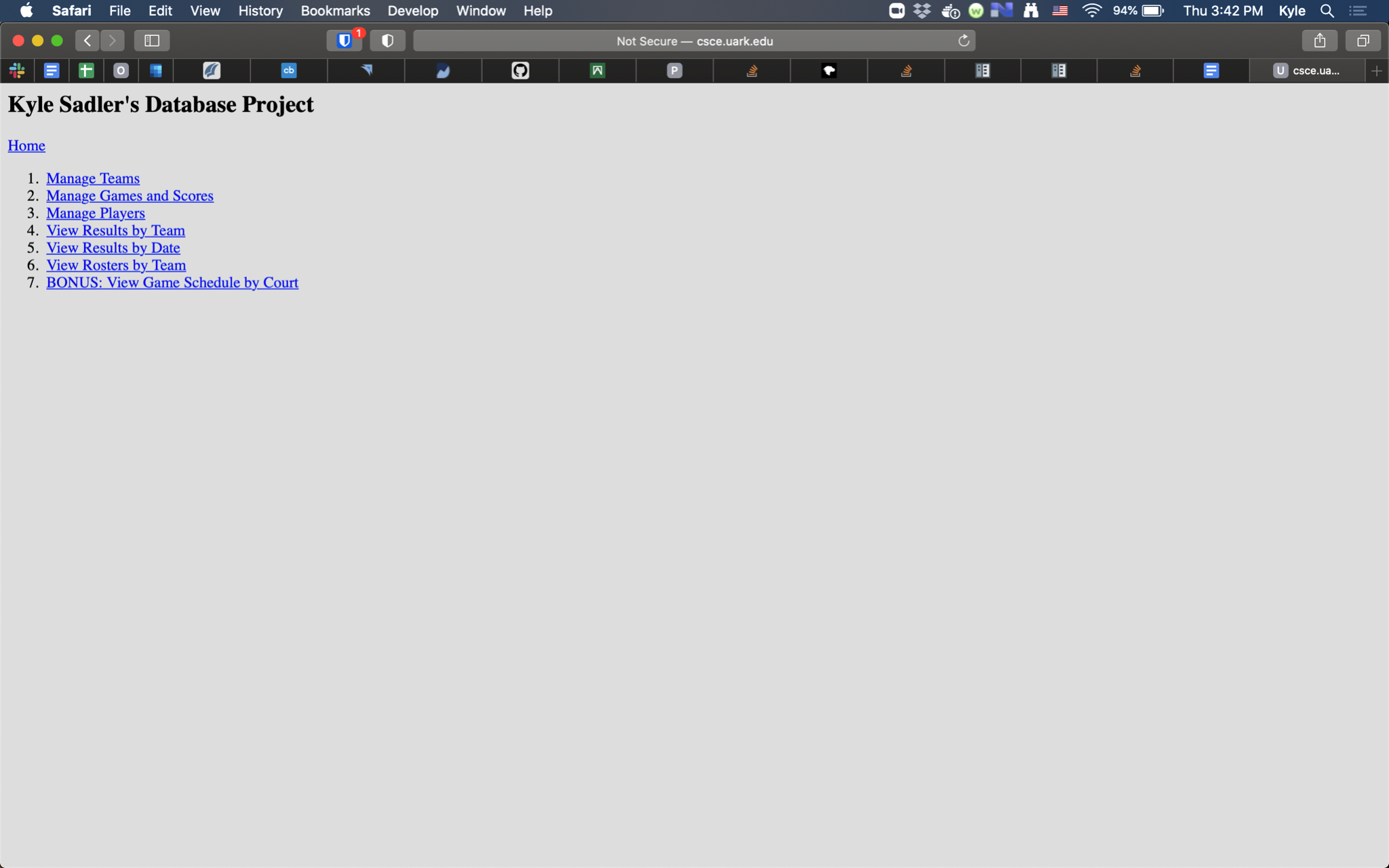
|  |  |
| --- | --- |
| **Function Page** | **Description** |
| Manage Teams | View all teams, add teams to the database |
| Manage Games and Scores | View all games and their scores, add new games to the database |
| Manage Players | View all players, add players to the database |
| View Results by Team | View all results for a selected team |
| View Results by Date | View all results for a selected game date |
| View Rosters by Team | View all players and their positions for a selected team |
| BONUS: View Game Schedule by Court | View all completed and scheduled games for a selected basketball court |

For function pages that required users’ unvalidated input, I wrote a utility function to clean user input before constructing any SQL queries. The cleaning process consisted of removing the characters . ' " ; ( ) ! = - which helped prevent the user from escaping from the constructed SQL query and injecting code into the database. If an error was thrown in the backend due to empty or invalid input, an error message was displayed to the screen.

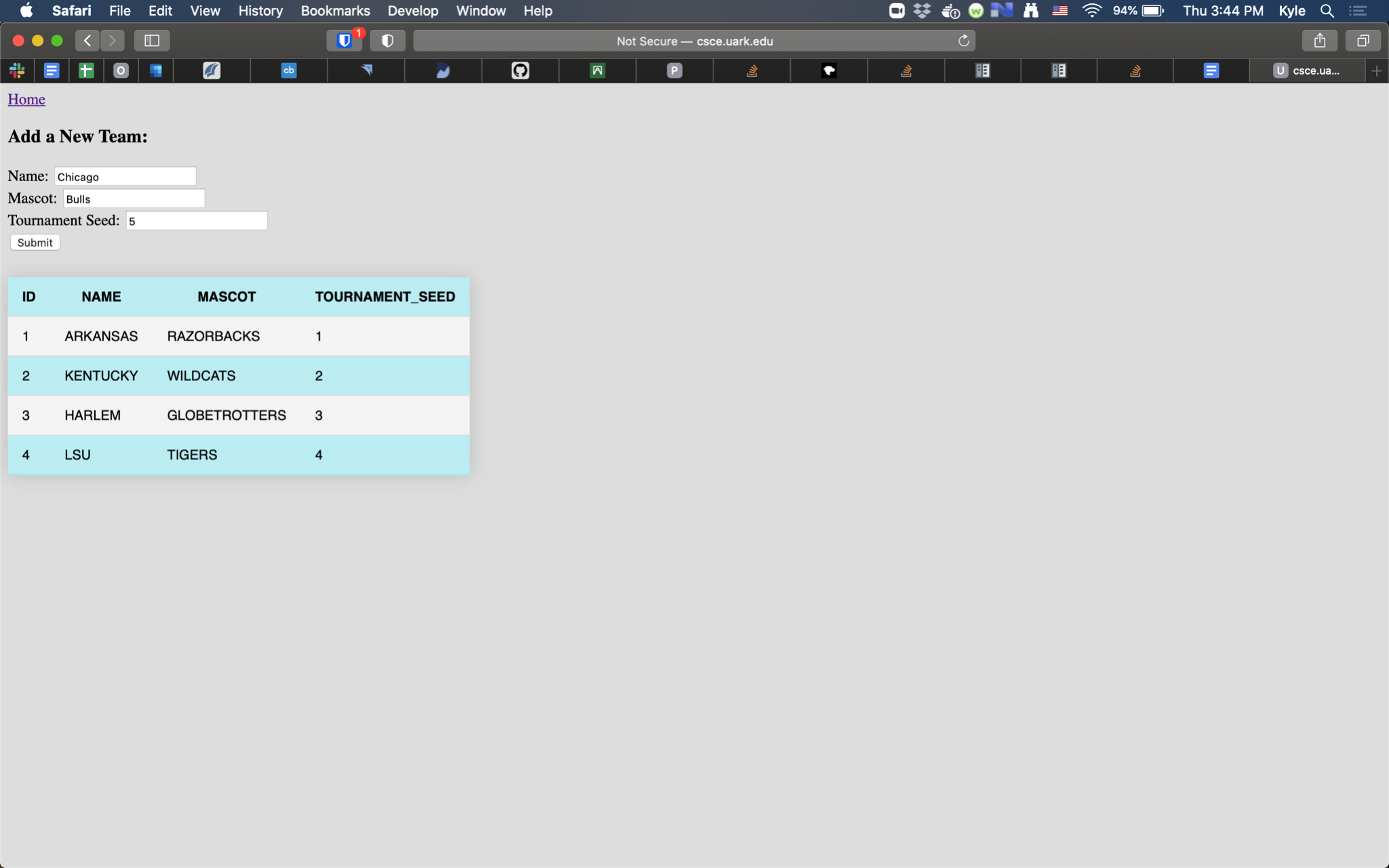
**Results**

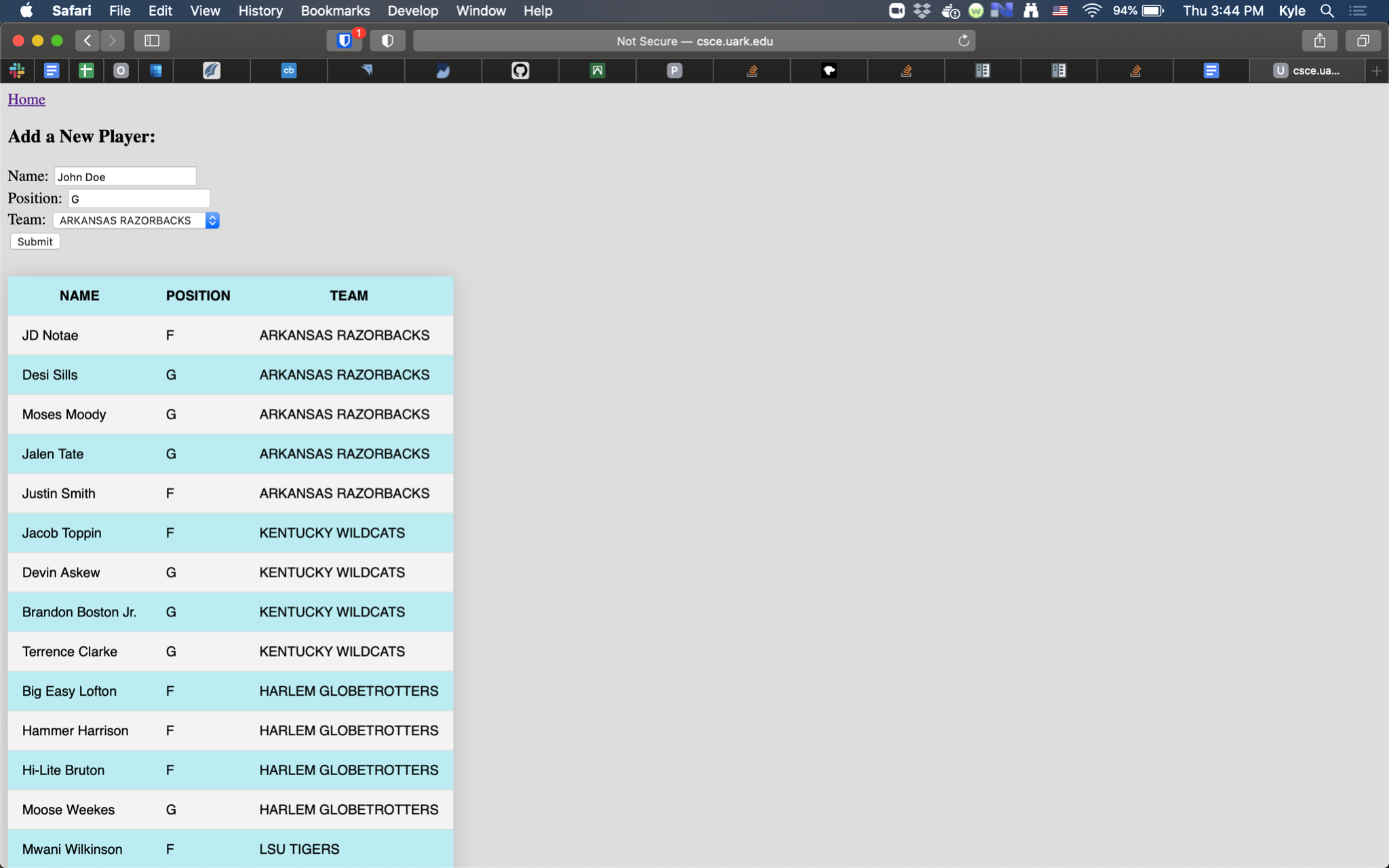
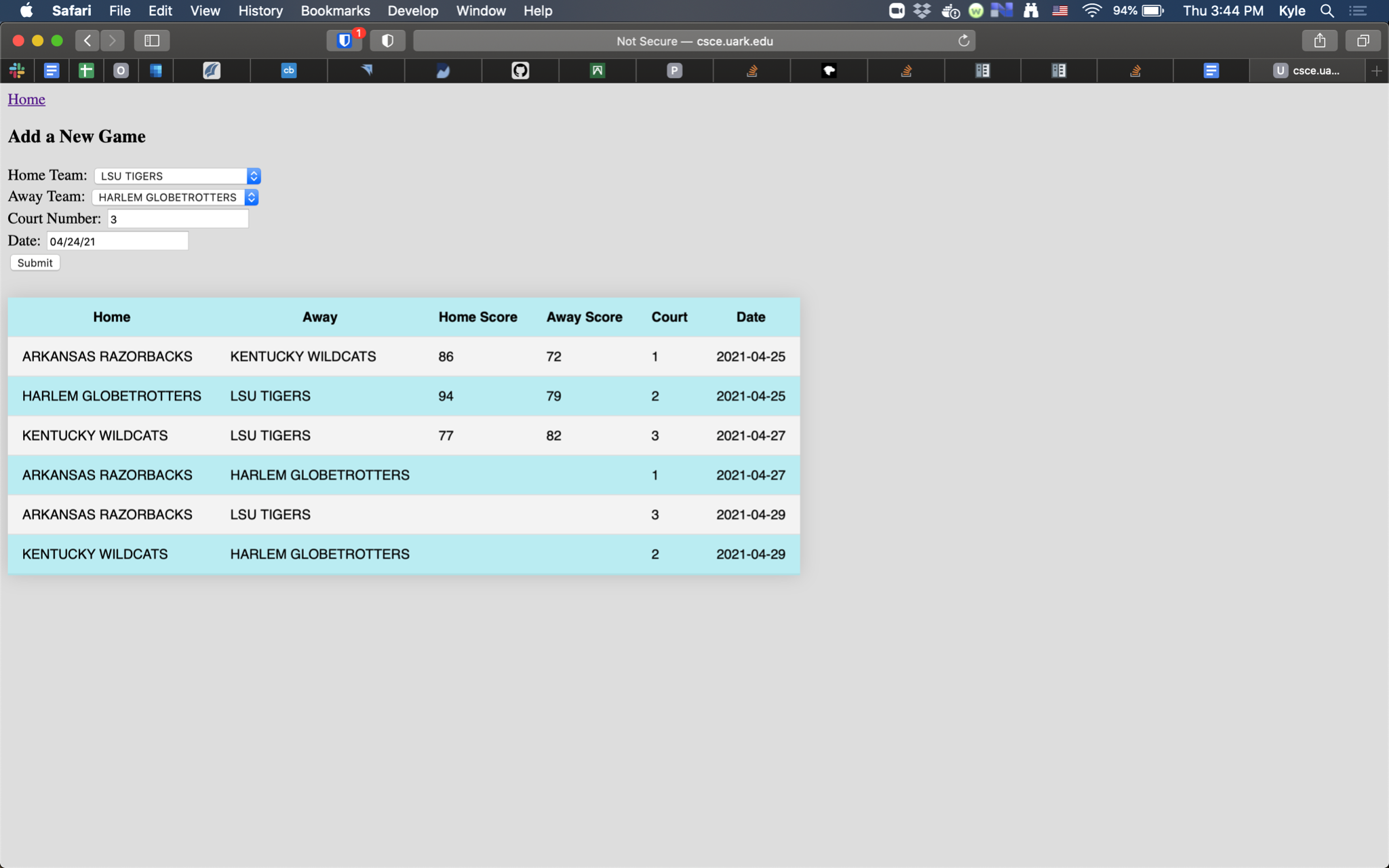
The result was a web application that allowed the user to interact and perform complex queries on a relational database. The design decision that worked the best was using JSON to pass objects between PHP and Python. This allowed me to call a function in PHP, pass arguments to Python, then pass the results back to PHP without having to parse or format anything. It worked like magic! I am also proud of creating PHP utility functions to parse arguments from POST requests and call backend Python functions by name. These functions were the bread and butter of this project and saved so much repeated code.

The result is live on <http://csce.uark.edu/~krs028/index.php> and screenshots of the pages are shown below.



**Figure 1.** Home page with links to each of the seven function pages.

**Figure 2.** The Manage Teams page allows the user to view all teams and add new teams to the database.

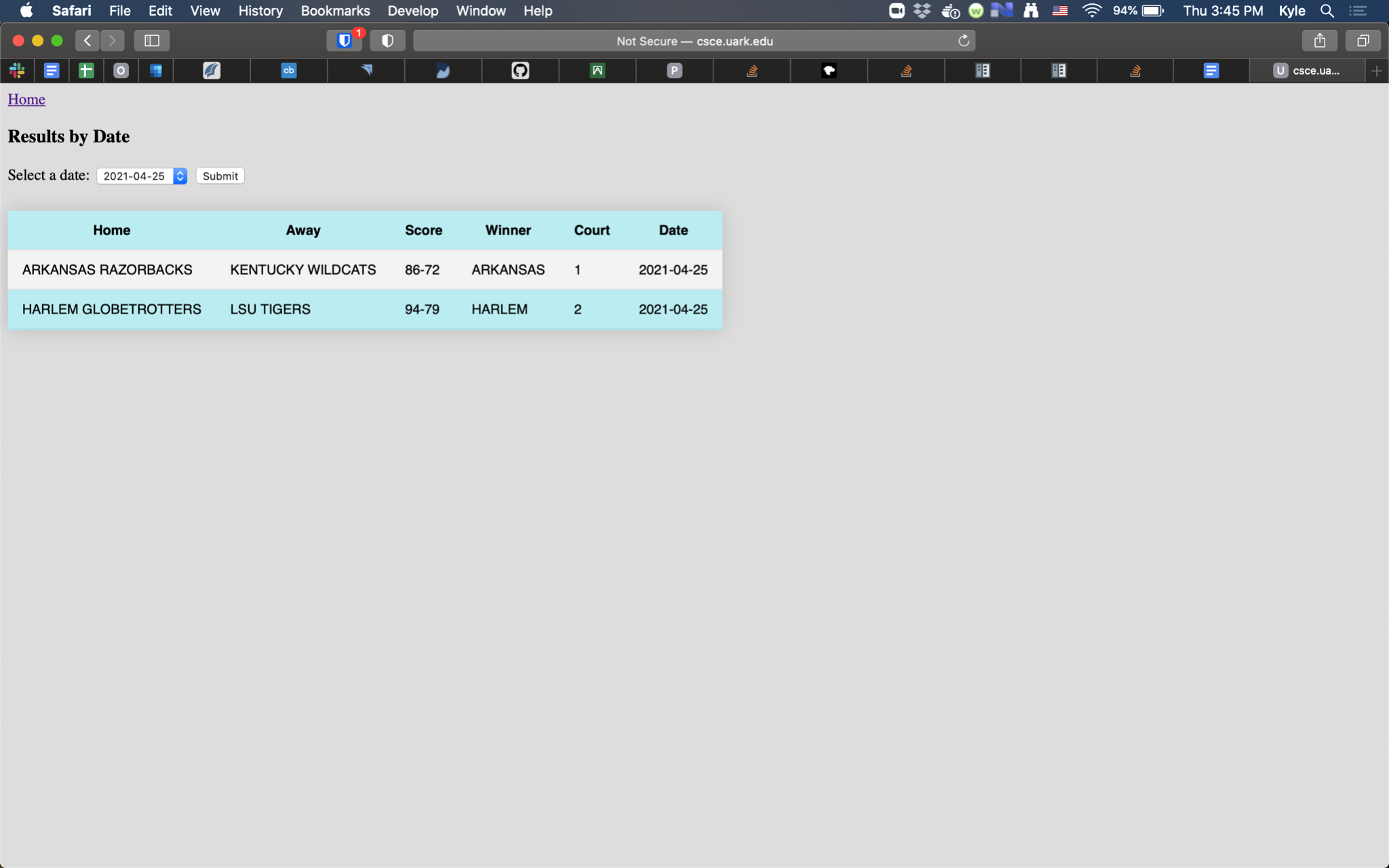
**Figure 3.** The Manage Games and Scores page allows the user to view all games, including the scores if the game is completed, and add new games to the database.

**Figure 4.** The Manage Players page allows the user to view all players in the tournament and add new players to the database.

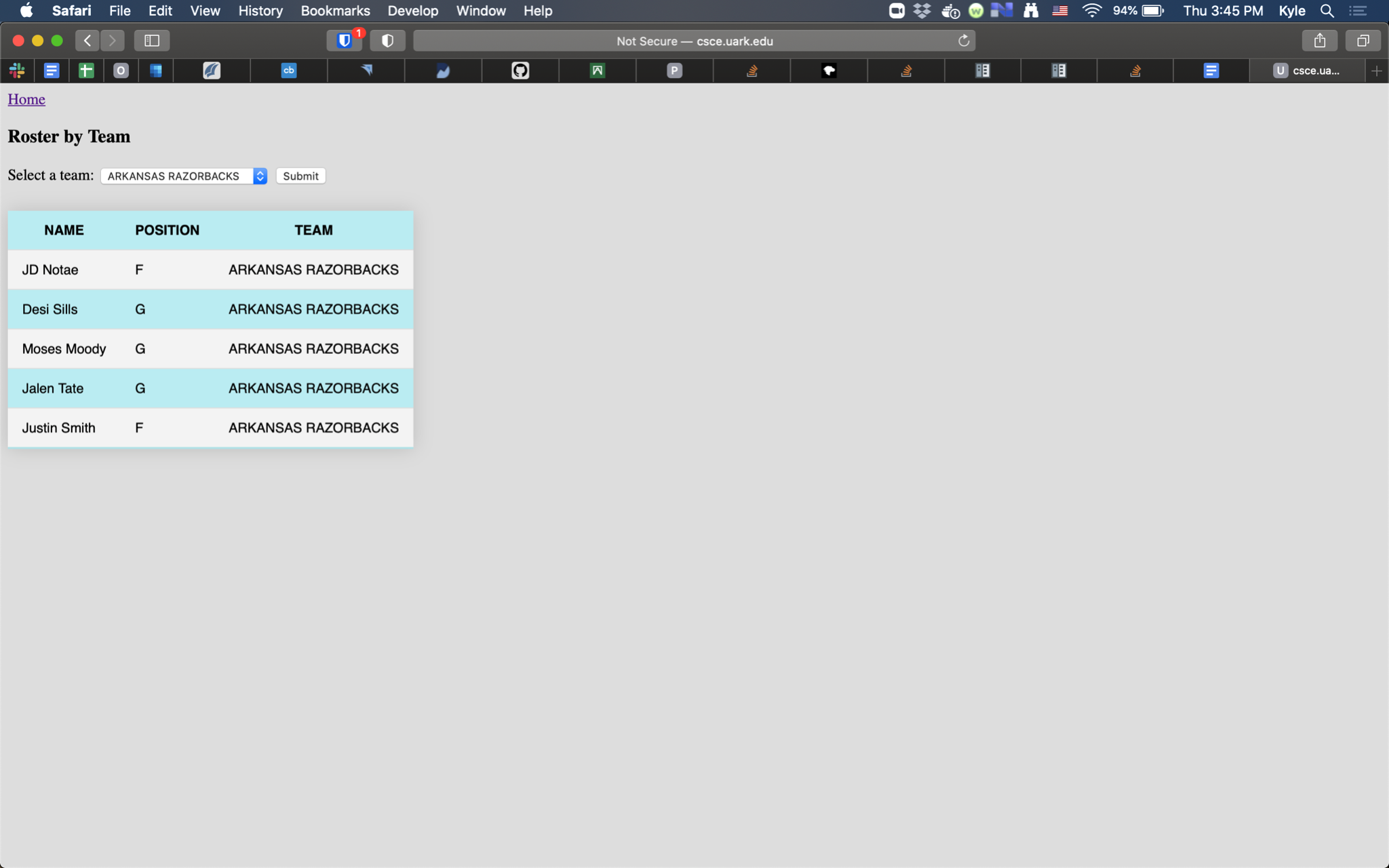
Graphical user interface, text, application

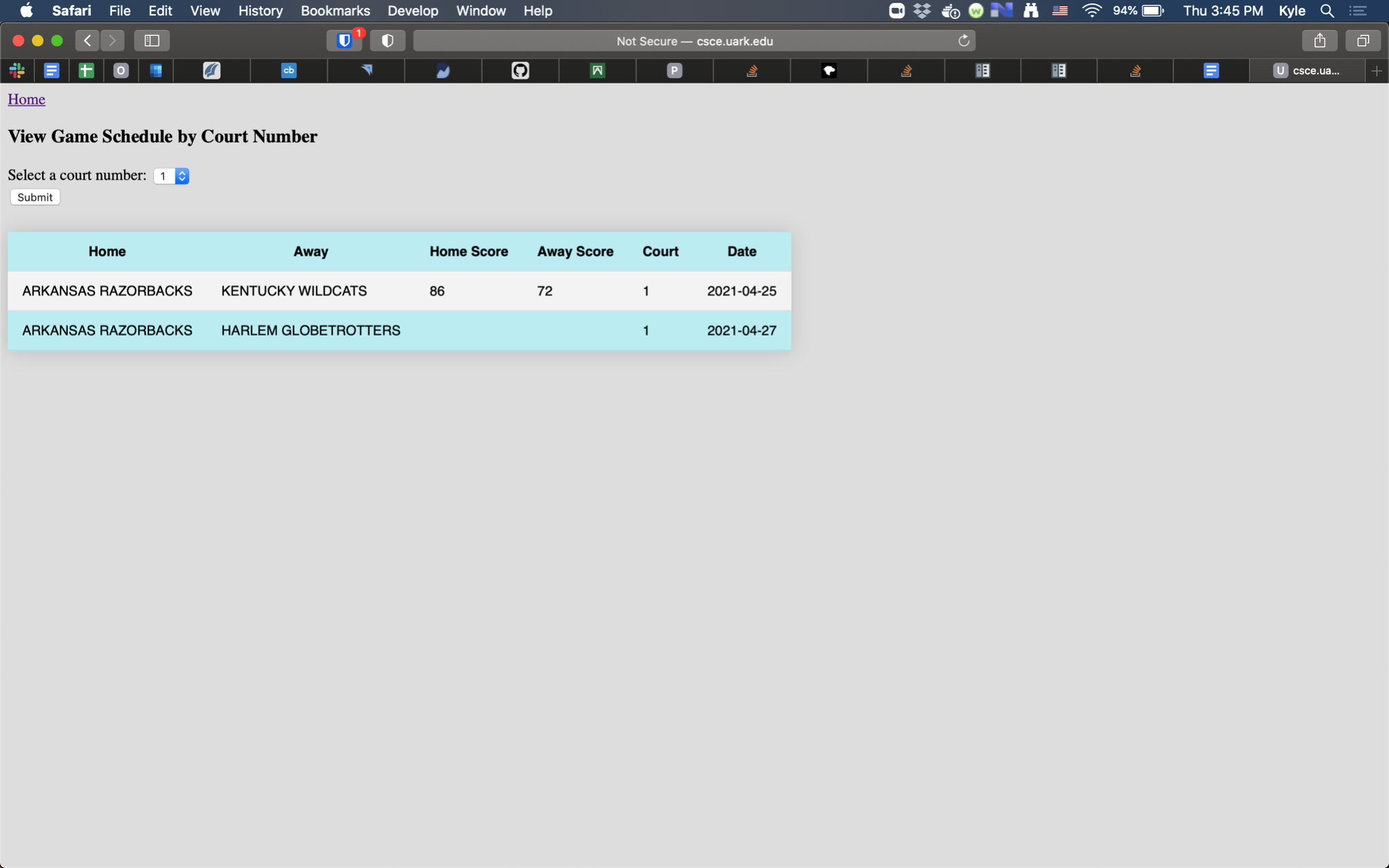
Description automatically generated

**Figure 5.** The View Results by Team page allows the user to select a team from the database and view all results for that team.



**Figure 6.** The View Results by Date page allows the user to select a date from the database and view all results from that date.

**Figure 7.** The View Rosters by Team page allows the user to select a team from the database and view all its players.



**Figure 8.** The Game Schedule by Court Number page allows the user to select a basketball court from the database and view all the games played on that court.

**Testing and Conclusions**

As I was developing the web application, I tested one page at a time. I first debugged each SQL call in MySQL’s interactive shell. Once my database calls were working, I tested calling my Python backend function from PHP. After I could reliably call my Python backend from PHP, I tested the user input on each webpage by entering valid and then invalid values. I made sure to check that each page displayed an error message when the user input was empty or invalid.

Overall, I am very happy with this project. I felt like the end product was robust and something I could be proud of. I completed this project in 15 hours over 3 days.