

Football Ticket Sales System Final Report

Zihengh(Zihengh@g.clemson.edu)

Chongm(Chongm@g.clemson.edu)

Problem Description and Motivation

Clemson Memorial Stadium contains over 80000 seats, without a platform to manage and display all this information, it will be hard for visitors view and buy tickets in sections they like. Meanwhile, users also need a portal to cancel their plan and apply for a refund while managers of the stadium may want to adjust the price for each section. We designed a ticket sales system to meet all these demands.

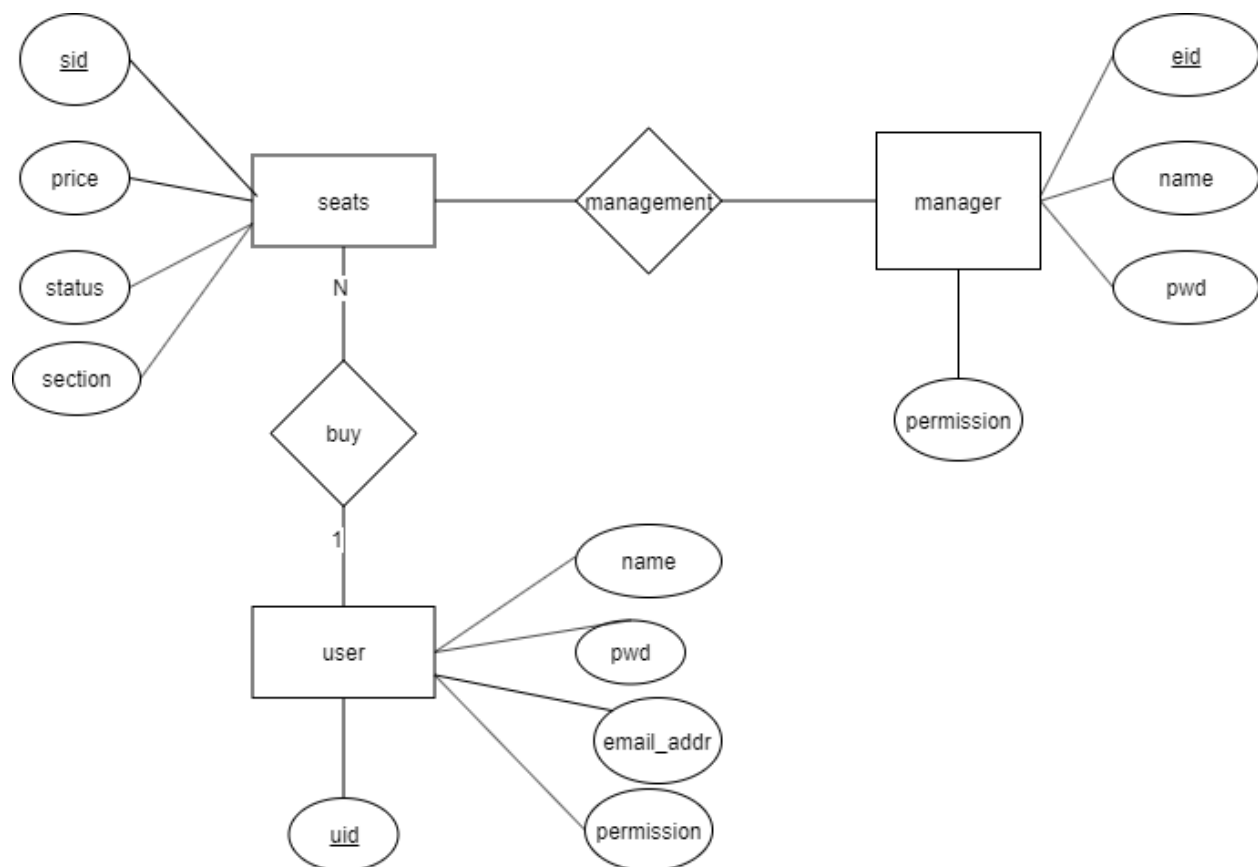


Figure 1

System Architecture

The final ER Diagram is showed in Figure 1.

To make it easy for users of any roles to use, we designed the function models and interfaces as Figure 2 showed.

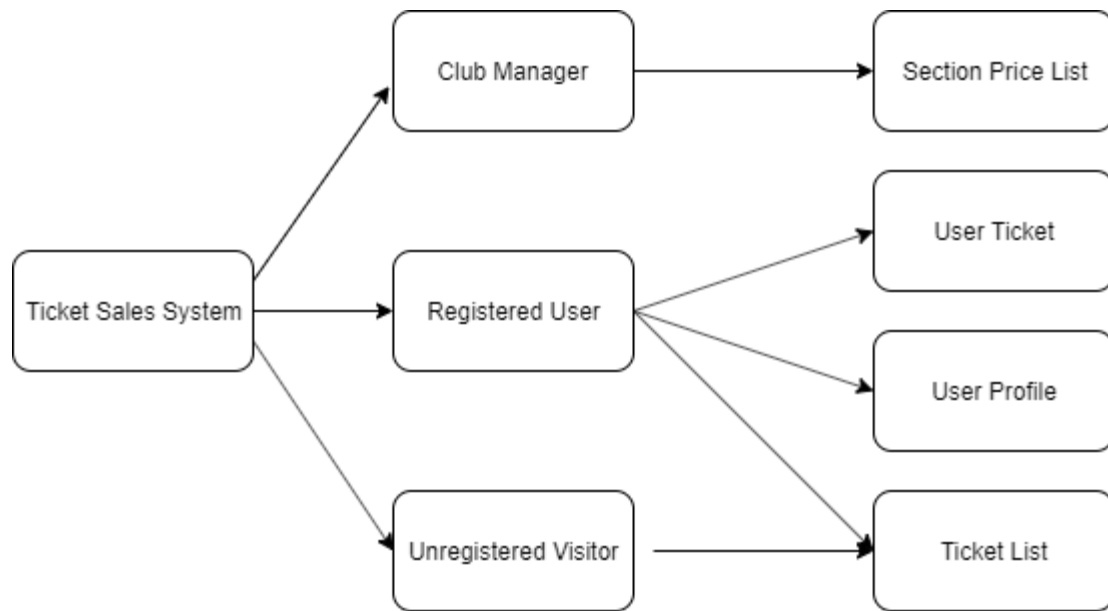


Figure 2

From the program view, the structure is like the Figure 3. HTML defines the display framework. JS files are responsible for connecting server, receiving/sending data, and dynamically draw the interface.

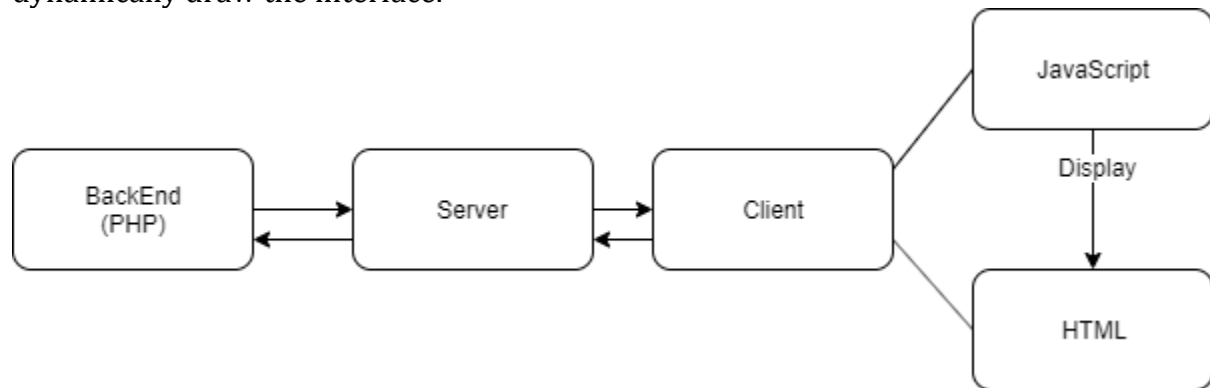


Figure 3

Function Implement

Figure 4 shows the first web page of our system. Visitors can view the price, number of remaining seats and price of each section either logged in or not. The following list describes **how to test** this page:

- The webpage provides two ways to filter the searching result. If number is put into the input box, the ticket list will return sections who have remaining seats more than that number. Once mouse is moved into a section in the stadium map right side of the page, the section will be highlighted. The list will return the specific section once clicked on.
- Once logged in, users can buy the exact amount of tickets in the section when they clicked on the 'Buy' button at the end of the section row. If not logged in, the page will jump to the signup/login page.
- Visitors can always jump to other pages to view the tickets they bought or modify their own profile through the tab link at the top.

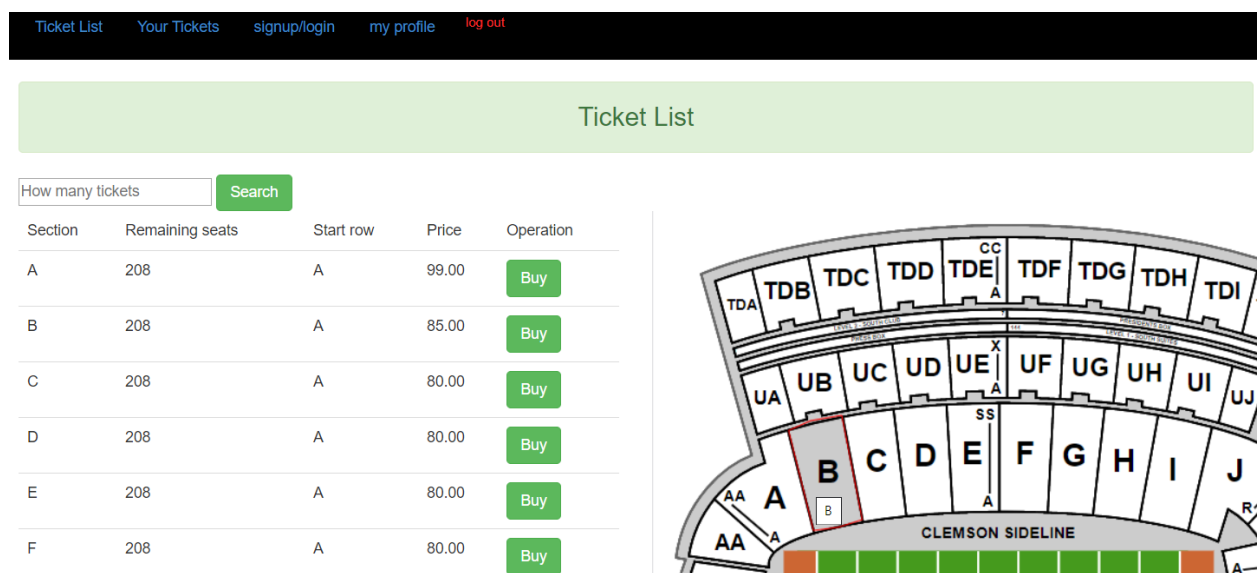


Figure 4

Figure 5 shows the web page for users to view their tickets and apply for a refund. The following list describes **how to test** this page:

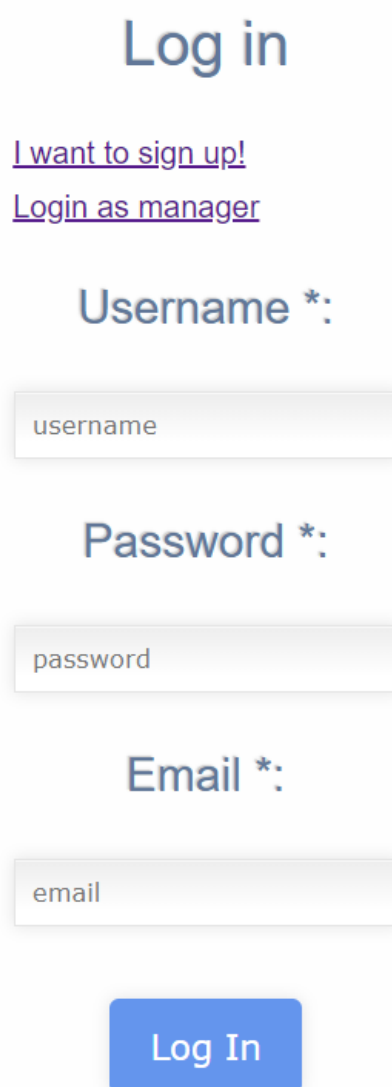
- The webpage will list all the tickets you bought once you are logged in. By clicking the 'Refund' button, the ticket will be deleted, and the corresponding change will be made to the ticket list page.
- No information will be listed if no user logged in the system. Page will be redirected to signup/login page once clicked on the 'Refund' page under this circumstance.
- Visitors can always jump to other pages to view the tickets they bought or modify their own profile.

View Tickets	Your Tickets	signup/login	my profile	log out
Your Tickets				
seats	Price	Operation		
A-A-1	99	Refund		
A-A-2	99	Refund		
A-A-3	99	Refund		

Figure 5

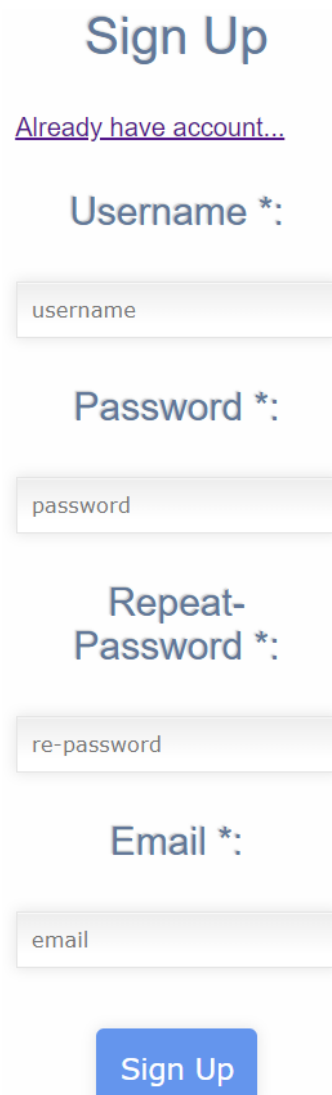
Figure 6 shows the web page for login. Figure 7 shows the web page for sign up. The following list shows **how to test** the login and signup function:

- Once successfully logged in or signed up, browser will jump to the ticket list page.
- We made certain constraints for signing up. Visitors can not register with a username that already exist. The password must at least contain a capital letter, a lower-case letter and a digital number. The register will not succeed if any of the previous constraints is violated. Information of these violation are shown by inner text like Figure 10 and Figure 11.
- Visitors can choose to login as a manager, once logged in successfully, browser will jump to the page designed for managers to adjust price for each section.



The login form is titled "Log in" in a large blue font. Below the title, there are two links: "I want to sign up!" and "Login as manager", both in blue text. The form contains three input fields: "Username *:" with a placeholder "username", "Password *:" with a placeholder "password", and "Email *:" with a placeholder "email". At the bottom is a blue button labeled "Log In".

Figure 6



The sign up form is titled "Sign Up" in a large blue font. Below the title, there is a link "Already have account..." in blue text. The form contains four input fields: "Username *:" with a placeholder "username", "Password *:" with a placeholder "password", "Repeat-Password *:" with a placeholder "re-password", and "Email *:" with a placeholder "email". At the bottom is a blue button labeled "Sign Up".

Figure 7

Figure 8 shows the web page for manager to adjust the price for each section. The following list shows **how to test** this function:

- The List shows the information of each section, including the number of remaining seats, the start row and price.
- Manager can make change to the price by typing new price into the input box. Once clicking on the save button, the web page will post the updated information and make update to database.

[log out](#)

Price List

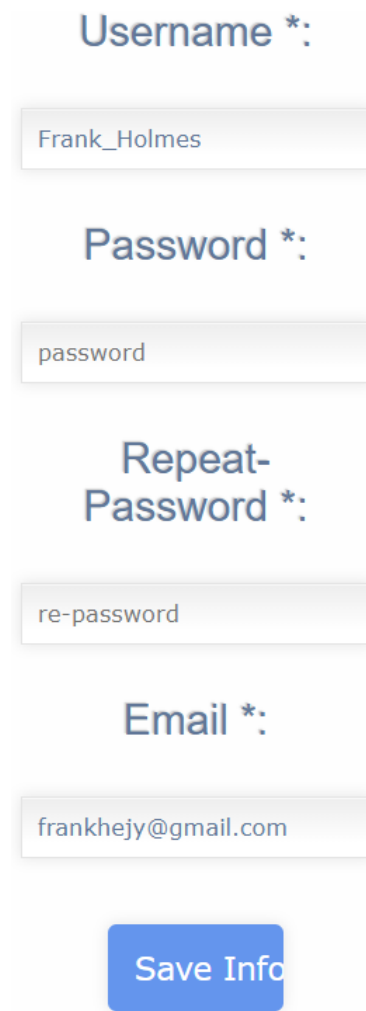
Section	Remaining seats	Start row	Price
A	204	A	<input type="text" value="99.00"/>
B	208	A	<input type="text" value="85.00"/>
C	208	A	<input type="text" value="80.00"/>
D	208	A	<input type="text" value="80.00"/>

Save

Figure 8

Figure 9 shows the web page for users to modify their profiles. The following list shows **how to test** this function:

- The original information of username and email address should be listed.
- We made certain constraints for signing up. Visitors can not register with a username that already exist. The password must at least contain a capital letter, a lower-case letter and a digital number. The register will not succeed if any of the previous constraints is violated.



Username *:

Password *:

Repeat-Password *:

Email *:

Figure 9

User name has been used,
please try another name.

Figure 10

Password should have at
least 5 characters.

Figure 11

Advanced function:

As Figure 12 shows, the admin can back up every data tables by clicking “Dump” button and they can recover data by pressing “Restore”.

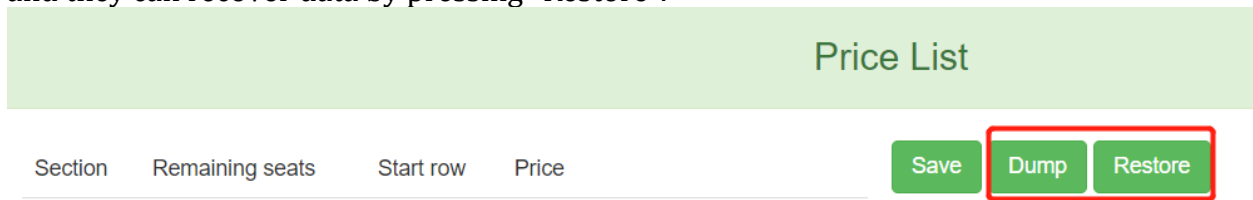


Figure 12

The future work

There is still a lot of space for improvement in our system.

- **About Cyber Security:** In our system, we did not design a Timeout Login Mechanism, we record the logged user information in PHP sessions, which means browser has forever authentication to specific resources once logged in. While recording the passwords, we only used MD5 conversion which is now vulnerable to certain attacking methods like rainbow table attack. We are planning to store the user's password in a better way and add a Timeout Login Mechanism to enhance security level.
- **About Transaction Atomicity and Consistency:** In our system, we maintain consistency by executing a series of MySQL queries. However, the consistency will be broken once one of the queries fails. We are planning to replace some of the queries by constraints and triggers. We will also add a rollback mechanism to deal with the situation when some queries fail to be executed.
- **About Transaction Efficiency:** When implementing some of the functions, for example, like adjusting the price for each section, we refresh the entire table when manager press the 'save' button, even if he/she just did a change to one section. In our observation, this process is time consuming and can be improved by adjusting the web page function.
- **About Database Dump and Restore:** There are three ways to dump and restore the database. The first is serializing the data to local file. The second is dumping and restoring the database via 'mysql' commands. The third is saving by record all the SQL queries to rebuild all the tables in a '.sql' file and restore the tables by re-executing all these queries. In our system, we choose the third one by which dumping is swift but restoring is rather slow. We are planning to replace this approach by executing 'mysql' commands.

The link to our system

<http://webapp.cs.clemson.edu/~zihengh/ticketlist.html>

Accounts

Manager:

Username: zihengh

Password: zihengh

Common user:

Username: Frank_Holmes

Password: Zihengh123