Game Contest Server

Juan Felipe Comboni
Department of Computer Science & Engineering Taylor University Upland, Indiana 46989

Introduction

Game Contest Server is a project that was created this past Fall Semester. The Game Contest Server's main purpose is to allow users to upload automated players and participate in various contests and challenges.

Game Contest Server:

This J-Term a group of Senior worked developing this original project which was started in the Fall of 2013. The projects was Done in Ruby 2.0 and Rails 4.0. I Used a platform that allowed me to code which is called nitrous.IO this platform uses colors just like the Terminal and allows you to code in many languages.

Our Goals:

- ✓ User Interface Redesign
- ✓ Admin Interface Functionality
- ✓ Continuous Test Integration
- ✓ Back-end Development
- ✓ Pagination and Search Functionality

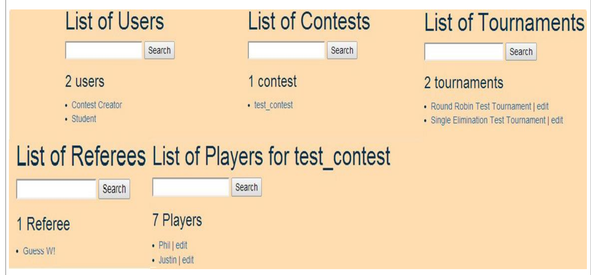

Figure 1: Search Functionality display on the website

Start

The Idea of having a search functionality in the website came in a discussion that we had as a group. As an Example of a similar instance look at amazon, searching it is Ideal that in this particular page because something that I have learned in my E-commerce class is that the less clicks a user does the better it is because users like doing the minimum work possible to the results wanted.

Search Functionality

Some of the front-end additions that was implemented in the website was a search functionality, which was added to the users, referees, players, contests, and tournament pages. The implementation allows the users to find content depending on which page the search box is used.

Implementing this functionality in the website was very interesting because three fundamental type of searches had to be implemented in order to have a simple search functionality in the front end of the website.

The first search feature that had to be supported was "full word search" meaning the results should display only the word that was entered).

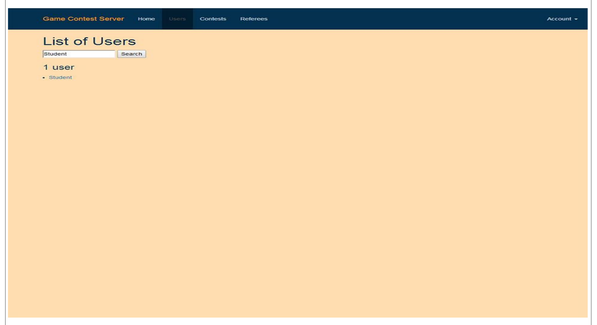

Figure 2 : Search Functionality Display of Full word Searching

This functionality is fundamental to the website because as web-developers, it should not only be efficient for users to navigate through the website but also it should be located in a specific visible place so it can be easy used by users.

The second search feature that had to be supported was "partial word search" meaning the results should display all the matching letters of the input entered.

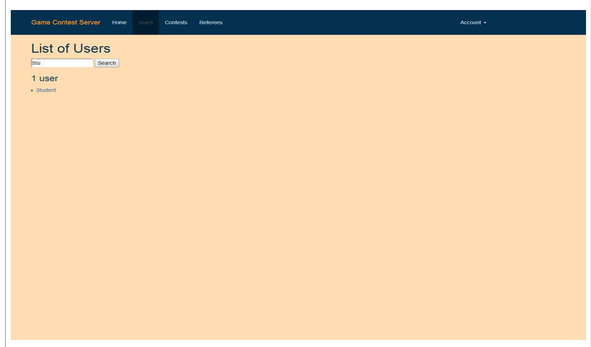

Figure 3 : Search Functionality Display of Partial Searching

Search Error Handling

Error handling in this functionality is important because as web-developers feedback is one of the most common and best way to communicate to the users that either something went wrong and they have to fix it or everything is running smooth.

The third search feature that had to be supported was "error handing" meaning the results should display a visible feedback to the user explaining the miss match search with the input and the results.

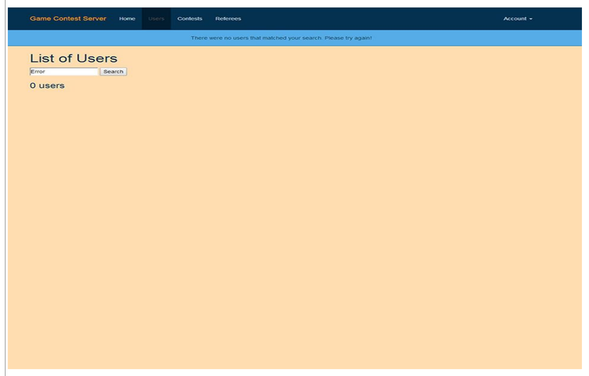

Figure 4: Search Functionality Error message display

Testing for this functionality was long but good because doing tests would not only help me know what functionalities I need to implement to make the simple search possible but also helped me assure the functionality was working properly.
Factory girl was the original implementation for testing the code of the original code that students started in COS243.
I also used Factory girl to make the search functionality test.

```
describe 'search_pagination' do
  let(:submit) {"Search"}
  before do
    FactoryGirl.create(:user, username: "searchtest1")
    FactoryGirl.create(:user, username: "peter1")
    FactoryGirl.create(:user, username: "searchtest9")
    FactoryGirl.create(:user, username: "peter9")
    FactoryGirl.create(:user, username: "searchtest3")
    FactoryGirl.create(:user, username: "peter3")
    FactoryGirl.create(:user, username: "searchtest4")
    FactoryGirl.create(:user, username: "peter4")
    FactoryGirl.create(:user, username: "searchtest5")
    FactoryGirl.create(:user, username: "peter5")
    FactoryGirl.create(:user, username: "searchtest6")
    FactoryGirl.create(:user, username: "peter6")
    visit users_path
    fill_in 'search', with:"searchtest4"
    click_button submit
  end
  after(:all)  { User.delete_all }
  it { should have_content("1 user") }
  it { should_not have_link('2', href: "/?commit=Search&direction=&page=2&search=searchtest4&sort=&utf8=✓" ) }
end

describe 'search' do
  let(:submit) {"Search"}

  before do
    FactoryGirl.create(:user, username: "searchtest")
    visit users_path
    fill_in 'search', with:"searchtest"
    click_button submit
  end

  it 'should return results' do
    should have_content('searchtest')
    should have_content('1 user')

  end
end
```

Figure 5: Testing code Sample

Most well designed websites make the Users take the least amount of clicks to navigate through the website so the users don't get frustrated. The search functionality in the game server content allows partial word searching meaning that it is able find words even though only a few letters have been entered.

Conclusion

This project has been a unique experience because I was not only happy I learned a new language which was ruby and rails but also I was able to implement test for the features that I added to the website. My experience doing this project made me develop skill that makes me say that I have coded and developed a website. I am very thankful for people that will use this software to their advantage.
The search functionality helped me understand how most websites on the internet work. From now on I will look to a website differently because not only I will appreciate the design and functionality but I will be able to understand all the functionality the website provides also I would be able to relate to the amount of work it takes to develop one.
This Functionality could be implemented to a more complex search tool that in the future could be useful. I believe that right now the search functionality works for the purpose of the website. The partial search allow the list to be sorted for easy access to the word desired.

Future Work

- ❖ Visually displaying games as they are played
- ❖ Improve error checking, such as if the player or referee dies
- ❖ Much more robust unit testing of the backend.
- ❖ Create Matches page with pagination and search
- ❖ Add download player functionality

acknowledgements

- Dr. Geisler for helping us debug and brainstorm
- Nate White and Nathan Lickey for helping us set up a VM\
- Dr. White and Dr. Brandle for help with python library and port communication