BeerMe

Ruby on Rails Application

Assignment 01 - HCI 432 - Mike Standeven

Overview

I created a website application that could store any number of beers entered along with a brewery, rating, note and a wishlist function. Each beer entered into the database included the brew (title), the brewery, the ranking (1-5), a user entered note, and a boolean wishlist mode (yes or no). I included a search function on the homepage and included a way for the search to be cleared.

Process

The BeerMe project was created using Ruby (v. 1.9.3p194) on Rails (v. 3.2.11). I generated a scaffold from the command line using Beer as the model. The classes I used were title (string), brewery (string), rating (integer), note (text), and wishlist (boolean).

Editing the ‘beer.rb’ model allowed for the validation of the beer title and brewery field as well as the selection of a numerical value (1 through 5) for the rating of all beer entries. Each view shows the wishlist boolean that was edited to show a ‘yes’ or ‘no’ instead of the default ‘true’ or ‘false’ value.

In the view, I edited the default table based layout and placed each beer into a div that floats to the left and fills the browser window in a grid format (with css edits in the assets folder). I edited the title of each beer to also be the link to the individual beer page. Each individual beer page has a ‘back’ link back to the home page. Each individual beer view shows an edit and delete functionality that does not appear on the home page list (which is the default view).

I created a universal graphic header by editing the application layout (application.html.erb) by creating an image header that is shared among all public facing pages.

The home page included a search function. The search option was created initially by editing the beers controller and defining a search function, editing the ‘beer’ route setting, and creating a search specific page showing the search results.

Observation

The current state of the application shares the search functionality via the search page and the index page; To make the application more efficient, I would remove the two step process of this function and make the index page control all of the search functionality, and remove the need for a search specific page.

An additional future state for this application would include individual users, user ratings, user notes and ratings of each brew, and a function for users to enter their own beers that are not already entered in the application.

Conclusion

This was a great intro application to learning Ruby on Rails. I have seen the basic potential of dynamic content creation that Ruby on Rails can provide. I look forward to incorporating more advanced features into my applications in the very near future.