Alan T. Grubb
Assignment 2

URL:

http://cs496-assignment2-993.appspot.com/

Description:

This website allows the user to create a 'Location' which consists of some basic data such as a Name, Active (boolean), an Image, a Star Rating, and a Description. Of attributes, only a name is required. The rest are coded as optional properties that can be stored for the location. However, due to the html code of the website for the submission form, most of the fields are required in order to submit a location.

This website will provide the basis for a larger web application that will make use of 'Locations' that can be added by users. I envision a web application that allows users to collectively organize pick up sports games at particular locations. In order to do this my web application will have to make use of a lot more objects and stored properties for those objects. This, assignment 2, is a crucial step in the direction of completing this web application as it illustrates the building blocks necessary to program the project.

Test Suite:

Test Case	Expected	Result (Pass/Fail)
Submitting a Location with a name.	Name for Location is correctly stored in datastore.	Pass
Submitting Location without a name.	Requires name before form is submitted.	Pass
Editing a Location and removing the name for submission.	Requires name before form is submitted.	Pass
Submitting a Location with spaces only as a name.	Requires a name to have >=1 character.	Fail
Submitting a Location with Active checked.	Active for Location is stored as True in datastore.	Pass
Submitting a Location with Active unchecked.	Active for Location is stored as False in datastore.	Pass
Editing a Location with Active checked.	Active for Location is stored as True in datastore.	Pass
Editing a Location with Active unchecked.	Active for Location is stored as False in datastore.	Pass
Submitting a Location with an image.	Image for Location is stored in datastore.	Pass
Submitting a Location without an image.	Requires image before form is submitted.	Pass
Editing a Location to remove image.	Image is removed for Location in datastore.	Pass
Editing a Location to not change image.	Image is unchanged for Location in datastore.	Pass
Editing a Location to change image.	Image is changed for Location in datastore.	Pass
Submitting a Location with a rating.	Rating is stored for Location in datastore.	Pass
Editing a Location with a rating.	Rating is changed for Location in datastore.	Pass
Submitting a Location with no description.	Description is not stored for Location in datastore. Location still created.	Pass
Submitting a Location with description.	Description is stored for Location in datastore.	Pass
Editing a Location to remove description.	Description is removed for Location in datastore.	Pass
Editing a Location to change description.	Description is changed for Location in datastore.	Pass
Navigating to edit page of a Location.	All fields are prepopulated correctly for that Location.	Pass
Navigating to view page of a Location.	All fields with values are displayed for that Location and cannot be changed.	Pass
Adding a new Location.	Location is displayed immediately without a refresh of the page.	Pass

Results of Testing:

All of my test cases that tested basic functionality of the application passed with the exception of one. A test case of storing a Location with a blank (only spaces) Name failed and results in some albeit low priority erroneous behavior. All my basic cases passed, as I expected, because I performed unit testing all along the way while developing. I would test basic functionality after implementing every feature of the web application. This made testing after completion much easier as I wasn't running into all sorts of unexpected behavior anymore. For the test case that failed, I know why it failed - because it is storing a string for the name that isn't technically empty as it consists of blank spaces. The fix for this would be trivial as I would just call or create a function to remove all spaces and replace them with some other character and then evaluate the input string for name. This way, it would catch names full of blank spaces because those spaces would have been removed resulting in attempting to store an empty string which would fail.

Templating:

I decided to utilize the jinja2 templating language to create my web application. I initially planned on using Djnago because I had seen it listed in job descriptions for web developers. I decided to look into the differences between the two and it appears that they are really quite similar now. When jinja2 was first released, jinja2 was markedly faster to execute than django and garnered a lot of followers because of it. As new versions of django were released, though, this gap dissipated and any more it is more a matter of preference on which you choose. I chose jinja2 because of the differences in performance in the past. Maybe jinja2 is still slightly more efficient?

I enjoyed using templates for this assignment. It really makes the source code a lot cleaner and keeps things much more organized for me conceptually. In brief, a template provides encapsulation for displaying the dynamic data for the application within the html skeleton. Since this can get hard to read and maintain quick, templates allow you to create essentially different view models for the website and keep their code separate and easy to come back to and change/keep up with. I had templates for each of my html pages for Add a Location, View a Location and Edit a Location.

Reflection:

If I had to redo this assignment I would spend more time on what data to store for each location and the attributes of those stored properties. I would implement comments for each location, whereby users would be able to make comments on each location and all comments for a Location could be viewed. I plan to do this in following assignments.

I think also, if I had to redo, I would have further utilized templates to have a base page for the web application. I do not like that my 'admin.html' and 'admin.py' contain code for adding a location. I would prefer this to be in files that indicate that Add Location behavior is implemented within. To this issue, I would recreate this assignment by making a home admin page that links to an Add Location page which contains the necessary code for doing such. This would provide more a more logical conceptual framework and it would be easier to keep straight in my head.

Apart from the above issues, I am pleased with my implementation as it sits. I am hopeful that as I continue to progress through this project that I will be able to come back to this code and notice a lot more improvements I could make. I think this is indication of growth as a programmer. As it sits now, though, I still have too much to learn about Google App Engine to notice and discuss many more improvements I could or would like to make.