Course 12-780
Advanced Python and Web Prototyping for Infrastructure Systems

Fall 2016

Assignment 5 (Total: 14 points)

Due: Friday, December 9th, by Blackboard by end of the day. For each day the assignment is late, a penalty of 10% of your total grade will be deducted.

Task 1 (14 points) *Search in the drawing management systems*

This assignment task will continue the drawing management systems that. Please use the attached source codes to start this assignment. Note: If you directly open this project in PyCharm, you will need to setup the project as a Django project with the interpreter that has Django installed.

a. Create a new Django application in the project. Name it ***drawingSearchApp***. Add this application to the current Django project (0.5 pt).

b. In the ***drawingSearchApp*** application, create a new python module named ***search.py***. Create a method in this module and name the method ***DrawingSearch(keywords)***. Keywords is a parameter that will receive a string containing keywords to search for the drawing information. (0.5 pt).

c. Write code in the ***DrawingSearch(keywords)*** method to parse the keywords string, which contains information in this format (3 pt):

"*attributeValue1*,*attributeName1*:attributeValue2,*attributeName2*:[valueLowerBoundary-valueUpperBoundary], …"

One example of the keywards string is like the following:

"Porter Hall,DrawingID:Port_001,ConstructedYear:[1990-2000]"

Basically, comma ',' is used to separate each keywords. Each keyword can either just be a value, or contains the attribute name followed by the value with the separation colon ':'. Value can be just a single string, or a range that is defined using a lower boundary and an upper boundary. The two boundary values are separated by dash '-', and enclosed by square bracket "[" and "]".

The task is to parse this string to acquire each search keyword. The string should be first split using the comma sign to acquire each keyword. For each keyword, first check whether the attribute name is provided or not. If not, directly print out the keyword in the console in this format: (using the example keywords mentioned above)

©2015 Xuesong Liu
Please do not copy or upload without written permission

The keyword #1 only has a value Porter Hall
If the attribute name exist, print out the keyword in this format:
The keywod #2 uses attribute name DrawingID, and has a value Port_001

For each keyword, also check whether the value is a single value or range. If it's a single value, directly print the value as mentioned before. If it's a range print out the values in this format:
The keyword #3 uses attribute name ConstructedYear, and has a ranged value with lower boundary 1990 and upper boundary 2000

d. Task 1.c provides the way that we can parse a keywords string to use for the drawing search. In this task, we will write code in the ***DrawingSearch(keywords)*** method to search in the database (7 pt):

First, write code to import the ***models.py*** module from the ***drawingManageApp*** so that we can use the database.

Second, implement the functions to use keywords to query the drawing records. Write code to get a ***QuerySet*** of all drawing records from the database. (1 pt)
Then for each keyword, this ***QuerySet*** should be filtered using the provided data.

For each keyword, if the value is a ranged value, this value should only be queried for the attribute ***ConstructedYear***. If there is no attribute name for the ranged value, please return a string "Failed". No further keyword needs to be queried. Otherwise, write code to filter the QuerySet using the lower and upper boundary of the value. (2 pt)
(Hint: refer to this link about how to do the less/larger than query in Django http://stackoverflow.com/questions/10040143/how-to-do-a-less-than-or-equal-to-filter-in-django-queryset)

If the value of the keyword is a single value. Do the following implementation:
If there is an attribute name, write the function to filter the records directly using this attribute. (1 pt)

If there is no attribute name, write the function to filter the records using all attributes. For example, the first keyword in the Task 1.c only has the value "Porter Hall". This value should be used to filter all drawing records that has "Porter Hall" as any attribute value. (**Note**: You will need to use the OR operator to check whether "Porter Hall" may match any of the six attributes. Please refer to this webpage to use the Q operator for the OR operation: https://docs.djangoproject.com/en/1.8/topics/db/queries/#complex-lookups-with-q-objects) (2.5 pt)

Finally, after query all keywords, return the records in the QuerySet in a JSON string and return it. This step is similar to the last task in Assignment 3 (0.5 pt).

©2015 Xuesong Liu
Please do not copy or upload without written permission

e. In the ***drawingManagementPage.html*** file, please add an input textbox with ID "searchInput" at the top of the page. Then in the ***drawingManagement.js*** file, add a new function ***searchInputChange()***. Link this function to the onKeyUp event of the textbox so that whenever a key is pressed in the textbox, the event will be triggerd.

   In the ***searchInputChange()*** function, get the text from the input textbox. Check if the text is an empty or not. If not, use a jQuery Ajax call to send the text to the suburl "***/searchDrawing/***". This Ajax call should expect a string value as the returned value from the Django server. If the string is "Failed", do nothing and directly return. If the string is not "Failed", it should be a JSON string returned from the server containing the drawing information. Use the returned drawing information to replace the IDs in the dropdown list so that it will only show the drawings that meet the queried keywords.

   In the ***views.py*** module in the ***drawingSearchApp*** application, add a new function ***searchDrawing(request)***. This function should receive the text sent from the Ajax call. Import the ***DrawingSearch(keywords)*** function and execute this function with the keywords. Then return a HttpResponse with the string value received from the ***DrawingSearch(keywords)*** function. Add the ***searchDrawing(request)*** function to a new URL in the ***urls.py*** module. (1.5 pt)

f. After these steps, the webpage should be able to query drawings from the database. Please try it out and see whether there is any problem with this design. Is there any improvement you want to make in terms of the format of query string, the user interaction in the HTML webpage, the approaches for querying different keywords, etc.? Please discuss your findings and propose new ideas if you have any to improve the software design. There is no lower or upper limits on how many issues you should find out or discuss. (1.5 pt)

©2015 Xuesong Liu
Please do not copy or upload without written permission