HW 5 Report

Name: Karthik Natarajan

USC ID: 5247024172

Email: karthikn@usc.edu

STEPS TAKEN TO COMPLETE SPELL CORRECTION:

- To implement the spell correction feature the first step of the process is to download Peter Norvig's spell correction program (php version) and include it to the index.php page.
- Next I created a dummy php program to call the correct function of the SpellCorrector program to create the serialized_dictionary.txt which increases the efficiency.
- Next a java file was written called Htmlparse.java to parse all the html files and create a big.txt file which will be used by the SpellCorrector.php program.
- The Htmlparse.java file used Apache Tika library imported as a jar, to parse and create the big.txt file.
- Next I implemented the spell correction feature in the main index.php file.
- The main step of Spell Correction was to call the correct function of SpellCorrector.php which would give the correct spelling of the misspelled query.
- I used a variable called $div in my php code which is true when the query is incorrect
- Thus before I start to display the results of my query I see if the $div variable is true and if it is, I give a suggestion to the user to query with the correct word.
- In order to deal with memory constraints I added a statement ini_set('memory_limit','-1') in the beginning of my code that allows my php program to deal with space requirements of the SpellCorrector.php code.

STEPS TAKEN TO COMPLETE AUTOCOMPLETE:

- The first step in the process of implementing the auto complete feature is to make the suggested changes in the solrconfig.xml file to redirect requests to "/suggest" handler.
- As soon as a word is typed in the input query box as ajax xmlhttpresponse is used to communicate with Solr's in built suggest functionality.

- Internally my php code makes and ajax request to the "/suggest" handler when a user starts typing in the query box.
- The response after querying solr is then encoded into JSON formation using JSON.stringify and JSON.parse methods.
- The array of suggestions returned are then parsed and displayed in the form of a drop down box with clickable elements.
- After a user enters a new word, the ajax code will only take the last word because the first word has already been corrected.
- The style sheet http://code.jquery.com/ui/1.11.4/themes/smoothness/jqueryui.css is used to display the autocomplete results in a drop down clickable form.

STEPS TAKEN TO COMPLETE TEXT SNIPPETS:

- The first step in snippet generation is including the simple_html_dom.php parses in my php code.
- I used regular expressions to get rid of special characters from the text
- After the text is obtained from the html file using file_get _contents function and it is exploded into an array
- Then I proceeded to check if the terms of the query are present in the text using string matching.
- If the query terms are not present I proceeded to check the title and description to find a match and if there was, display the text associated with it.
- If the document title was not found at all, I went on to display "No Snippets".

ANALYSIS OF RESULTS :

Spell Correction:

1) bitcion -> Bitcoin
2) sugestion -> Suggestion
3) heatlh -> health
4) camerea -> camera
5) bakteria -> bacteria

Auto Complete:

1) ca -> canonical, catch, canceled, careers
2) t -> taxonomy, tech, terms, text
3) al -> all, alexa, also, alternate
4) de -> delivery, device, description, desktop
5) ga -> game, games, gave, garden