University of Toronto
CSC326: Programming Languages

Final Report

Green Light Search Engine

Group#:26
Jiaqi Tian 1000263546
Si Yi Wu 1000430759
Yan Wan 1000287511

1. Describe the design of your enhanced search engine in detail. If you enhanced an algorithm, describe the different candidate algorithms and how they are different from the baseline implementation in Lab 3, and describe the quantitative metric you use to judge the merits of the candidates and how you chose your final candidate.

In Lab3 we have implemented:

- Pagination for search results
- Size fitting after search implemented to align search box and logo
- Page ranking of urls in backend
- Multithreading in backend to increase performance during page ranking

In this Final Lab, we have enhancement of following features:

- Spell Correction
    - Add a file with a list of words dictionary
    - Check every input word and correct them with the most probable existing word
    - Do searching based on the corrected words
- Autocompletion
    - Add a list of dictionary for autocompletion during inputting
- Multi-word Searching
    - Can accept multiple words as input
    - Can search based on all input words
    - Eliminate duplicate urls
- Simple Math Calculation
    - Can take math calculation string as input
    - Output result and the math equation in page
- Complex Ranking System
    - Make use of damping factor to increase the accuracy of page ranking
- Optimize Search Data Structure
    - Make use of numpy array to faster access data comparing to list
- Minimize Number of Clicks for User
    - Show search box every page, so extra click to query page not needed

2. Brief (no more than one page of text) high level documentation of your project's code, including where all features (i.e. pagination is implemented in file_x.py , page ranking is implemented in files a.py and b.py) are implemented, any external dependencies and how the different files relate to each other (i.e. high-level UML diagram).

- Front End
    - Pagination is done in both FrontEnd.py and views/Search.tpl files which Search.tpl is a template to which we pass the pagination information
    - Display of search engine logo and search box is done in views folder Equation.tpl, Search.tpl, Search_page.tpl files
    - Spell correction feature includes FrontEnd.py and words.txt, words.txt is the dictionary of words

    - Autocompletion is done in Equation.tpl, Search.tpl, Search_page.tpl files, they all have a list of words to do autocompletion
    - Multi-word Searching is in FrontEnd.py
    - Simple Math Calculation is done in FrontEnd.py, and display using Equation.tpl template
    - Search.tpl display the initial search page, Search_page.tpl display the search page with search results with pagination, Equation.tpl is used to display simple math calculation result
- Backend
    - Complex Page Ranking algorithm and multithreading is implemented in pagerank.py
    - Loading and storing crawling data from/to sql database and crawling web information is done in Crawler.py, and pagerank is instantiated in Crawler.py
    - The one click deployment is implemented in aws_deployment.py. Within aws_deployment.py, aws user credentials are read from a separate credential file, and then an aws instance is created. After that, the script also changes the redirect url line in the FrontEnd.py. The script then makes a call to aws_setup() function.
    - The setup of the aws instance is realized in the aws_setup.py. The aws_setup() function installs the required libraries on the aws instance, and then runs the FrontEnd.py to bring up the search engine and keep it running in the background.

3. Indicate the difference of your proposed design and completed design if there is any. If the search engine is completed differently than the proposed design, explain why.

There are three major enhancements we also proposed to implement in addition to the existing ones, but was not able to. They are listed and explained below:

- "Next Page" / "Previous Page" button
  During Lab 3, for pagination we also planned to add the "Next Page" and "Previous Page" button to jump back and forth between pages. However, we failed to implement this functionality due to approaching deadline and lack of time.
- Aesthetic Appealing
  We also thrive to improve the aesthetic appealing of the search engine. However, the previous code implementation structure and styles makes it extremely difficulty to improve the visual effects. We decide to stick with the "simple is the best" theory.
- Addition of GIF
  We also plan to add a small GIF icon to the search engine page, however, we were not able to find an appropriate GIF that suits the theme of our search engine, and decide to leave out the idea.

4. Explain your testing strategy during the development. Describe how you identify the corner cases.

- Unit testing: Testing each unit on a single function level.
- White box testing: Internal functionality testing by controlling inputs and examine for expecting outputs.

- Integration testing: Preparation of system testing, which means ensuring that each of the component is working on a proper working linkage. Apart from themselves individually functioning inside the whole system.
- System testing: Testing all of the system as a combination of components.
- Corner cases: Using false data types and limit cases to test out the functionalities(like empty string/list/dictionary, corner values and boolean values)

5. Lessons learned from this project.

- Expect from unexpected. Another course due could lead to limited workstations available. Some remote user on the workstation may significantly take over available cpu computability for compiling. There are some groups' EC2 accounts phished so they couldn't finish testing. Things actually could happen and make the real situation a lot messier than expected before.
- Limited time, limited probability. Having many ideas could be inspirational, but with actual time frame there are times that decisions has to be made. Certain proposals as to be given up and make room for more feasible and executable proposals.
- Learning takes time.For a project with many new technologies inside a new tech stack, climbing the learning curve could cause significant delay beside proper actual scheduling.

6. Describe what you would do differently if you had to do it again. What would you do if you had more time. Did any parts take longer than you thought, and Why?

- Thing would do:
    - Can use beautiful GIF as logo
    - Aesthetic issue on user interface, can improve the display in pagination page, display urls ,page numbers and add previous/next button on the bottom place of the web page, and display logo and search box on top of the web page to make it more user friendly
    - In autocompletion, can use more comprehensive ways, such as use the same dictionary of words as auto correction does
    - Can have search engine do complicated math calculation and plot some function waves
    - Can have screenshoot of highest rank pages, and display their web titles
- Thing takes long than thought:
    - Aesthetic part definitely takes a lot more time than expected, since it requires implementation of front end features as well as display logo, search box, search results in a good way. HTML, CSS, JSON programming skills are needed for doing this part

7. How the material from the course helped you with the project.

- Tutorials that covered some of the tools being used in the project(web framework)
- Python data storage method in lecture helps in backend implementation
- Data structure covered in lecture materials contribute to the work of backend page ranking and front end urls display

8. How much time it takes for you to complete each lab outside the lab sections.

- 15 (hrs/person)

9. Which part of the project you think is useful and you believe the labs should spend more time on it.

- EC2 manipulation, deployment and scripting

- UI design / interactive interface design
- Crawler implementation
- Pagerank algorithm and enhancement

10. Which part of the project you think is useless and you think it should be removed from the labs when this course is being offered in the future.

- HTML/CSS tutorial: The tutorial isn't quite helpful since it's very high level and not applicable to each individual project.
- Python basic syntax: This part can easily be self learned, and may be omitted from the course to save time for other meaningful and interactive teachings.

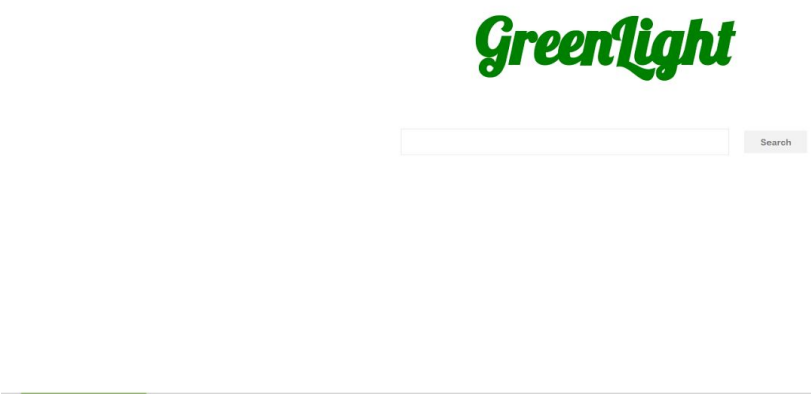
11. Other feedback or recommendations for the course.

- We need more hand-on instructions on how to combine what we learn to what we need to implement.
- We need more relevant design process training instead of plain syntax teaching.
- We need more responsible and knowledgeable TAs to lead for both offline and online supporting.

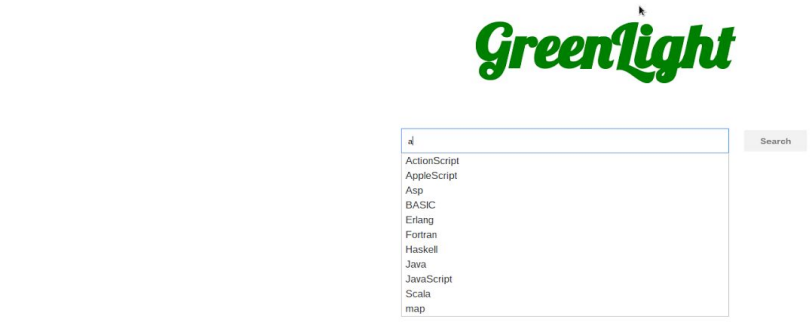
12. Responsibilities of each member. If you believe that workload is distributed unequally in your group, you may describe the situation in this section.

- Each group member made great contribution to the lab work, everyone is happy to collaborate and help each other in terms of development. And workload distribution for each member is listed:
    - Jiaqi Tian: Frontend design, documentation
    - Siyi Wu: Backend design, documentation
    - Yan Wan: Frontend design, documentation

Appendix A: Features Display



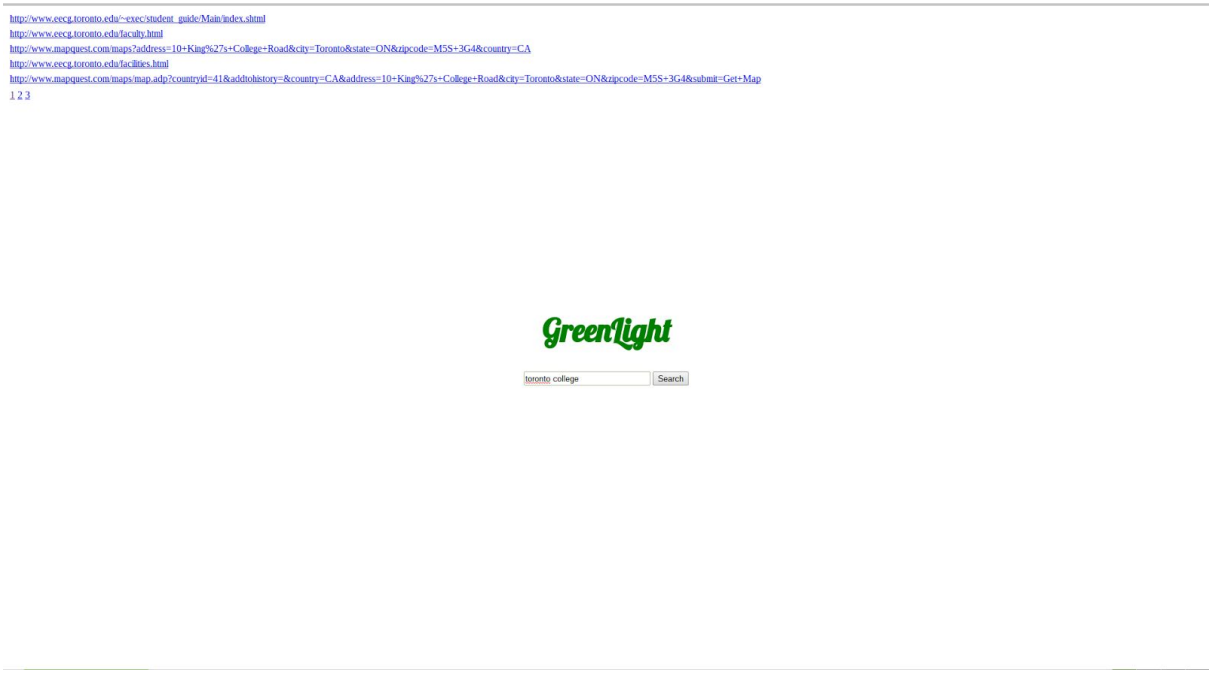
Picture 1: Initial Page of Search Engine



Picture 2: Autocompletion when typing input

Picture 3: Pagination

Picture 4: Simple Math calculation output



Picture 5: Multi-word Searching