**CSC326 Final Report**

December 2018

1. **Group member**

Jiehao Liu

Menglu Li

1. **Design of the search engine**
   1. Frontend

The search engine provides a query interface to the user and takes a single keyword that user types to the backend. In respond, the frontend page shows all URLs that related to this keyword in sorted order. Each page shows five URLs. User can view next five URLs by clinking page number or next page button.

* 1. Backend
     1. Crawler

The crawler algorithm visits all the URLs stated in url.txt, and each URL is assigned a page rank using PageRank algorithm. All information received by crawler are converted to string data type and stored in Redis database.

* + 1. Server

Server receives the keyword that user inputted and fetches out related ULRs from Redis database. Server returns these sorted ULRs back to the frontend.

1. **Features implemented in Lab4**

Most features we implemented in lab4 are enhancing the frontend display and user friendliness.

* 1. Spell check

The engine supports the feature of spell check and correction. It lets the user know when the words are misspelled by underlining the word and suggests the correct spelling to the user while the user is typing into the search box.



Figure 1 Feature of spelling check and correction

* 1. Search suggestion

Our engine has a search suggest drop-down list for users. Before the user finish typing query into the search box, a drop-down list with several suggested completed queries occurs to select. This feature helps the user to complete the search quickly.

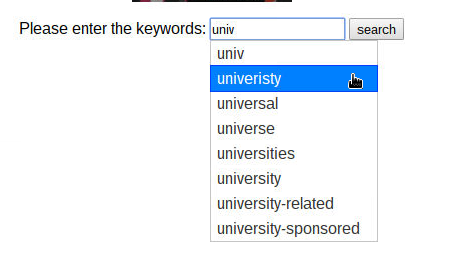


Figure 2 Feature of search suggestion

* 1. Minimum number of clicks required from the user

After finishing typing the query to the text box, the user can start the search by simply pressing the enter key without clicking the button on the screen. Also, the search box stays on the webpage, so the user does not need to click any button to start a new search.

* 1. Animated logo

In lab4, the logo of our search engine keeps the same theme as before, but changes to an animated one, which is catchy.

* 1. High responsibility

The webpage detects the user’ screen size and provide an optimal viewing experience across a wide range of devices, from computers to mobile phones or tablets. The page will resize its content to fit the size of display which the user chooses.

1. **Testing**

Our testing strategy for this project is running tests after each modification at the code to check whether each implementation is functioning and trying to track the bug step by step.

For example, by typing several different words to test the usage of the engine, we hit a corner case: when the search keyword was “home” or “google”, the search engine went down and returned a 500 error message. In order to troubleshooting, we printed out all content of PageRank and Lexicon dictionary which were stored in the database and realized that we reached a segmentation fault.

1. **Lessons learned**

This project provides us an opportunity to explore some new tools. This is the first time we learn to use Amazon Web Service to develop and deploy applications and websites. We also experience the Redis, which is a NoSQL database, for the first time. Furthermore, we are more familiar with the usage of Git abilities, since it is a teamwork which requires us work simultaneously.

1. **Modification and improvement**

If we have more time for this project, we plan to improve the algorithm of ranking system, which can advance the performance of our search engine, and add some more features to this engine, such as supporting multi-word searching. We also want to design a better user interface.

Implementing Redis database takes more time than we planned. As a NoSQL database, the data stored inside Redis is string type data, so we need to convert them into several dictionaries in Python which the frontend code can use.

1. **Useful material from course**

The course provides the background information about Python, such as the usage of dictionary data type. This knowledge is useful because both of us have not learned or practiced coding by using Python before.

1. **Working hours**

Each lab takes almost 15 hours in total to complete outside the lab sections.

1. **Useful part**

The idea of implementing a search engine is interesting. We believe the useful and helpful part of the project to us is giving us the freedom to try lots of new tools. For example, the lab instruction does not state explicitly which database we must use in the lab, then we can choose the one that is most suitable for our implementation.

1. **Useless part**

Forcing us to use aws cli command or boto package to access Amazon Web Services seems useless, since AWS provides a clear and easy-to-use graphical console to manage instances. Also the boto2 package which the lab instruction asks us to use is slightly out of date.

1. **Feedback**

Overall, this course is very interesting, and this open-ended project gives us an opportunity to think outside the box and try to relate the theory and concepts to practical implementation.

1. **Member Responsibilities**

Jiehao: Implementation of backend algorithm, also includes most frontend features such as enabling google sign-in and showing search engine result.

Menglu: Modification on frontend code, adjustment on interface design, deployment of application on AWS, unit testing, benchmarking and documentation.