CSC326 Lab 4 Report

Team Members:
Gligor Djogo, djogogli, 1000884206
Boyowa David Ogbeide, ogbeide1, 999644693
Gwyneth Bassett, bassettg, 1003696538

2. Describe the design of the enhanced search engine / difference between proposed design and completed design

Our design is a sleek and user friendly search engine. It allows users to login using their gmail accounts and search using the easy to use searchbar, available and visible on all pages of the site creating an efficient and user friendly searching experience.

Basic mathematical operations (+-*/) on integers will be evaluated and presented to users on the results page.

The search functionality will show results for multiple words. This is done by increasing the page rank of each page displayed according to how many words are related to that particular page, so the resulting pages are ordered by rank and relevance to the search phrase. Here is a pseudocode representation of the algorithm:

```
Set results = {init empty}
For (each ith word in sentence) {
      Gather results for the following individual word;
      Additively boost ranks of pages containing words[1 to i-1];
      Add new pages to the set of results (set intersection);
}
Sort(results) by rank;
Display(results) normally;
```

These features were not in the original design but were added to improve on our search engine.

3. The difference between the proposed design and the one implemented is listed above.

4. Testing Strategy:

In order to validate functionality, we would do "incremental testing" i.e. pass arbitrary data into the platform or execute a run-through of what has been coded so far to know that everything works. This way we can know exactly where the bugs are. To catch corner cases, we would choose best case and worst-case scenario situations for various variables and input in order to verify full functionality and flexibility.
For benchmarking, we would vary the ranges of input loads and concurrency to know the limits of the server and to know where weaknesses and strengths were in the platform.

5. Lessons Learned:

We have learned that every new software engineering project comes with many surprises in which initially we will have no idea how to do unless we do extensive research, tinkering, and analysis. Learning new APIs, integrating foreign frameworks, and dealing with multiple computing systems of varying types, all involve a hurdles – some of which are quick/easy but hard to find, and others which are time consuming but less hidden. The real world is not simple, sometimes things must be done outside the job description a.k.a “a puddle of liquid will spread as wide and as messily as it needs to until it is evened out or contained”.

6. What would we do differently? What would we do if we had more time? What took longer than expected, why?

If we had more time we would have focussed more on efficiency of the algorithms in the backend and added a few more features like spell correction and user-specific word suggestions to the front end. we would also automate the backend to gradually populate and update the database on the go.

On the front end the used authentication and routing using bottle took longer than expected. This was because we were unfamiliar with bottle and google APIs so had to read a lot of the documentation in order to implement them correctly. Pagination also took longer than expected.

Running an instance of the application using AWS was another unexpected delay because it takes upwards of 15 minutes to initialize and the virtual machine had a myriad of framework issues and permission blocks (gcc versions behind, linux issues, missing necessary packages for running python, using sudo thus cementing root privileges just for python to work).

7. How course material helped with the project:
Dictionaries, list comprehension, persistent storage, and classes helped with the search indexing and storage. Most of the knowledge and skills for the project came from the internet though.

8. How much did it take to complete each phase outside of the labs?
We spent two thirds of our time outside of the already scheduled lab sessions for this project.

9. Which part of the lab is useful and the the labs should spend more time on?

All aspects of the lab were useful. It was interesting to learn about routing, user authentication and using AWS while also using the python skills we learned in class. One of the most useful parts of the lab was the search indexing and database creation. More time should have been spent on teaching front end development and clearly explaining back end technicalities though.

10. Which part is useless and should be removed?

The useless parts of the lab were: registering our credit card numbers for a free service with the risk of unexpected charges (happened to another group), and a lot of technicality that wasn't part of the course nor overlapping with taught course material. We recommend to either narrow the scope of the project, or to preferably expand the scope of taught course material.

11. Other feedback and recommendations

More organization of course progression and labs. Some things were unclear, vague, and late.

12. Responsibilities of each member:

Backend, web server, and testing: Gligor and David
Frontend : Gwyneth