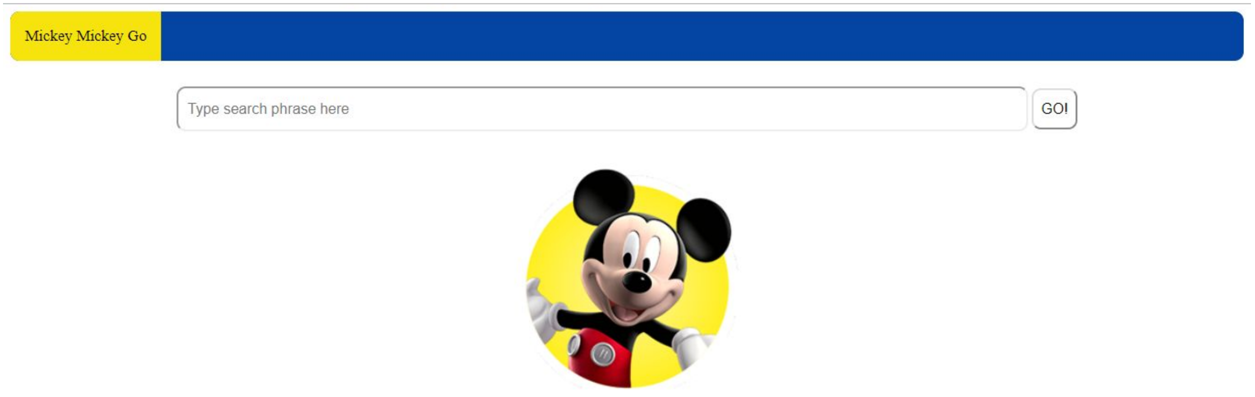
CSC326 Programming Languages

Lab 4 Final Project

Girija Khandekar (999856534), Wen Jie Feng(1000820766)



1) Names and Student numbers of all members

Girija Khandekar	999856534
Wen Jie Feng	1000820766

2) 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 metrics (i.e. benchmarks, profiling tools) you use to judge the merits of the candidate and how you chose your final candidate

For our search engine enhancement, we focused on feature route and implemented features to complement the frontend, aesthetics, user friendliness and the backend. Below are a list of features and the descriptions.

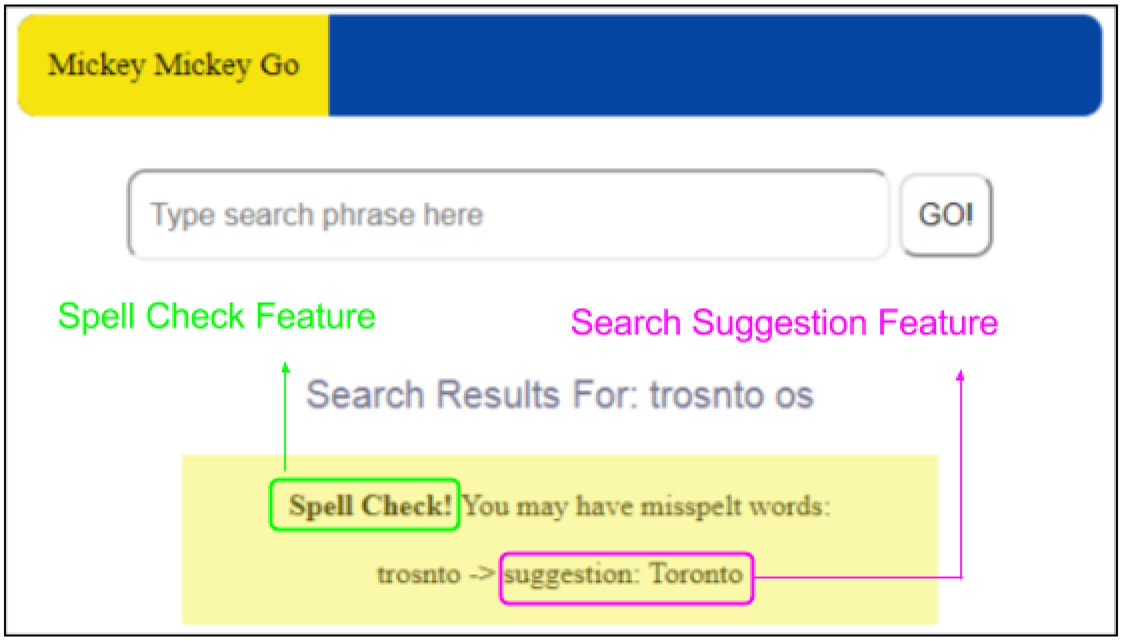
1. Spell Check

If one or more words is misspelled, the spell check feature is used to give the user feedback on the misspelled words. In this feature the user's input is broken down into individual words by splitting using spaces. Then the words are checked against the dictionary and the eecg website. If the word does not exist in either the dictionary or the website, it is considered as a misspelled word and flagged in the front end. Figure 1 below outlines the spellcheck feature.

2. Search Suggestion:

The search suggestion feature is tied with the spell check feature. The suggestion feature suggests a word to search based on the misspelt word. For example if a user inputs 'trosnto', the spell check feature will flag it as misspelt and the search suggestion feature will recommend 'toronto' as it closely resembles the user's input. Figure 1 below outlines the search suggestion feature.

Figure 1: Spell Check & Suggestion Feature



Search Results For: trosnto os

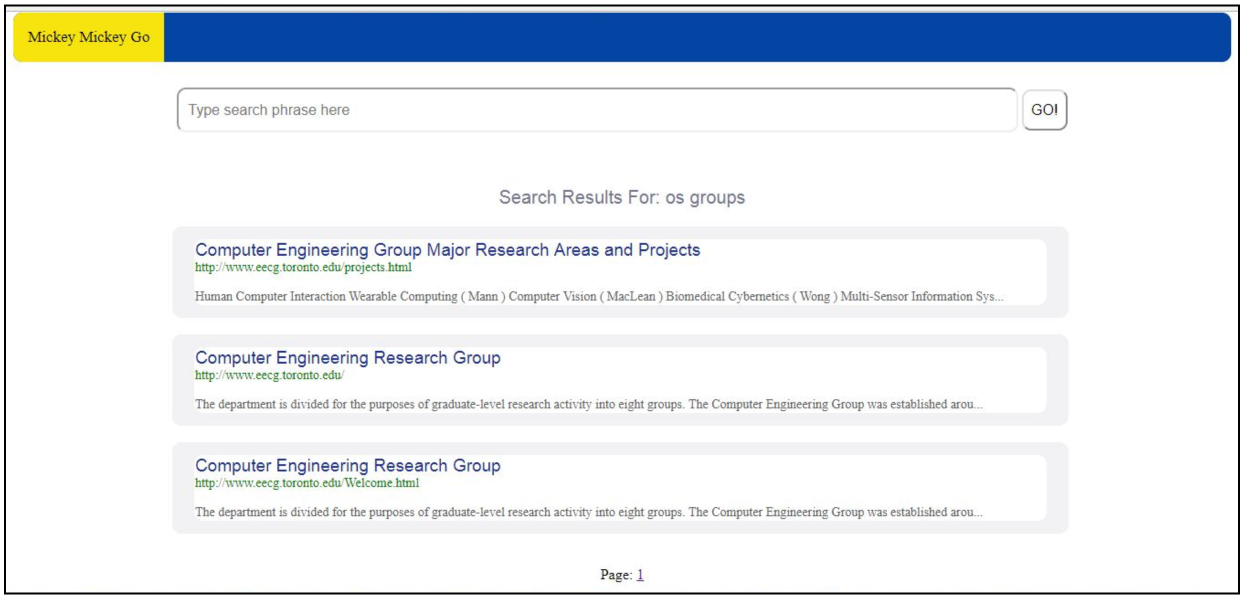
Spell Check! You may have misspelt words:

trosnto -> suggestion: Toronto

3. **Multi-Word Search:**

Search results are obtained based on all words in the user's input. Note that multi word search feature is also optimized to ensure that any words that are repeated do not result in the same links being repeated. For example, if the word 'os' is input by a user it results in only one link, likewise if the user inputs the phrase 'os os os', it will still only give one link since it's the same word is being repeated. Figure 2 is a screenshot which demonstrates the result of the phrase 'os groups'

Figure 2: Multi Word Search Demonstration



Search Results For: os groups

Computer Engineering Group Major Research Areas and Projects
http://www.eecg.toronto.edu/projects.html
Human Computer Interaction Wearable Computing (Mann) Computer Vision (MacLean) Biomedical Cybernetics (Wong) Multi-Sensor Information Sys...

Computer Engineering Research Group
http://www.eecg.toronto.edu/
The department is divided for the purposes of graduate-level research activity into eight groups. The Computer Engineering Group was established arou...

Computer Engineering Research Group
http://www.eecg.toronto.edu/Welcome.html
The department is divided for the purposes of graduate-level research activity into eight groups. The Computer Engineering Group was established arou...

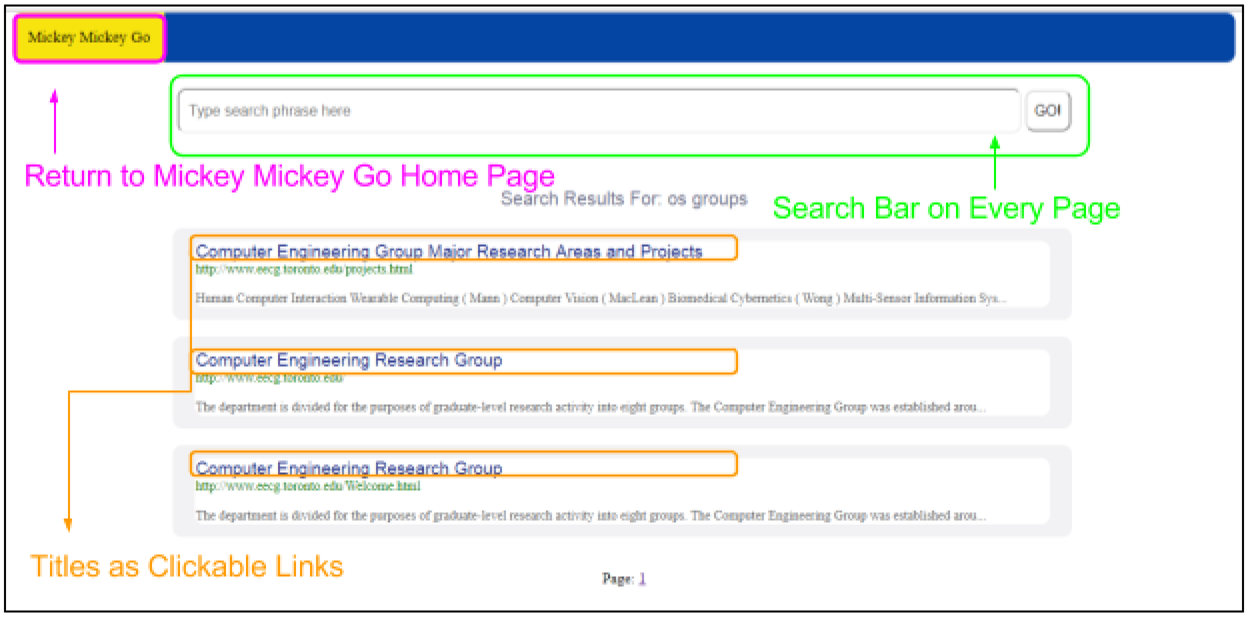
Page: 1

4. **User Friendliness**

Mickey Mickey Go search engine is optimized for user friendliness. One way this is accomplished is by displaying the search bar on every result page. By doing so, a user will not have to click the back button to do another search, rather they can continue searching while being on any of the results page. The option to return to the search engine's home page is also provided to the user, if a user wants to return to home, they can do so my clicking the yellow Mickey Mickey Go logo. Finally, the titles of all results are served as links to the page themselves. This allows the user to quickly go to another webpage without having the need to copy and paste the link in green.

Refer to figure 3 for user friendliness functionality .

Figure 3: User Friendliness Explained



Search Results For: os groups

Computer Engineering Group Major Research Areas and Projects
http://www.eecg.toronto.edu/projects.html
Human Computer Interaction Wearable Computing (Mann) Computer Vision (MacLean) Biomedical Cybernetics (Wong) Multi-Sensor Information Sys...

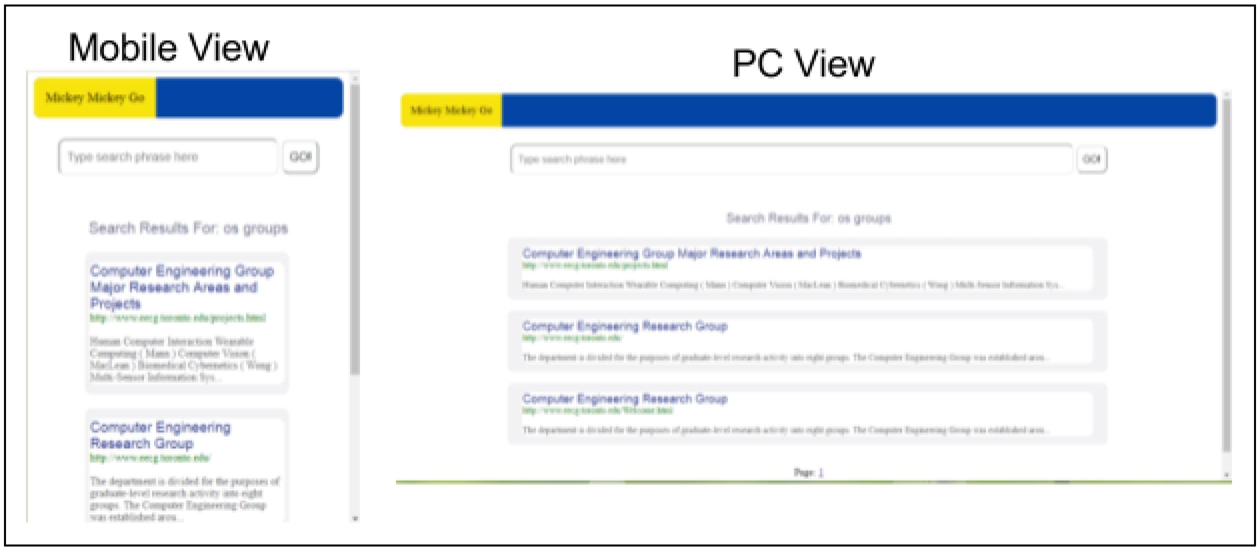
Computer Engineering Research Group
http://www.eecg.toronto.edu
The department is divided for the purposes of graduate-level research activity into eight groups. The Computer Engineering Group was established arou...

Computer Engineering Research Group
http://www.eecg.toronto.edu/Welcome.html
The department is divided for the purposes of graduate-level research activity into eight groups. The Computer Engineering Group was established arou...

Page: 1

5. **Customize for PC's, Tablets, Mobile Devices**
   The Mickey Mickey Go search engine adjusts to the screen size of the device. Thus as the screen is enlarged or made small the images and the search results will adjust accordingly. This functionality is maintained throughout the website. Figure 4 shows two views of the website.

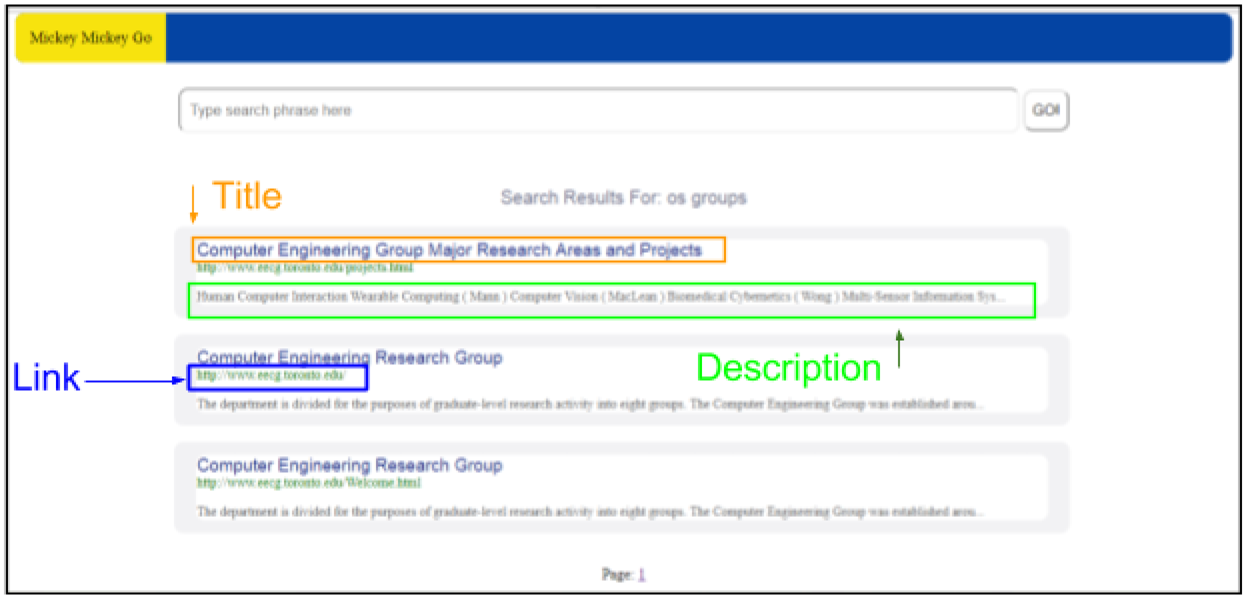
Figure 4: Responsive Web Page - Mobile View vs PC View



6. **Backend - Obtain Title**

The backend crawler code was enhanced to obtain the webpage title for a given url. This is outlined in the figure below. We believe that the title serves as a necessary quick summary of a given web page and therefore thought it was an essential feature to implement.

Figure 5: Obtaining Title and Description of Each link



Search Results For: os groups

Computer Engineering Group Major Research Areas and Projects
http://www.eecg.toronto.edu/projects.html
Human Computer Interaction Wearable Computing (Mann) Computer Vision (MacLean) Biomedical Cybernetics (Wong) Multi-Sensor Information Sys...

Computer Engineering Research Group
http://www.eecg.toronto.edu/
The department is divided for the purposes of graduate-level research activity into eight groups. The Computer Engineering Group was established arou...

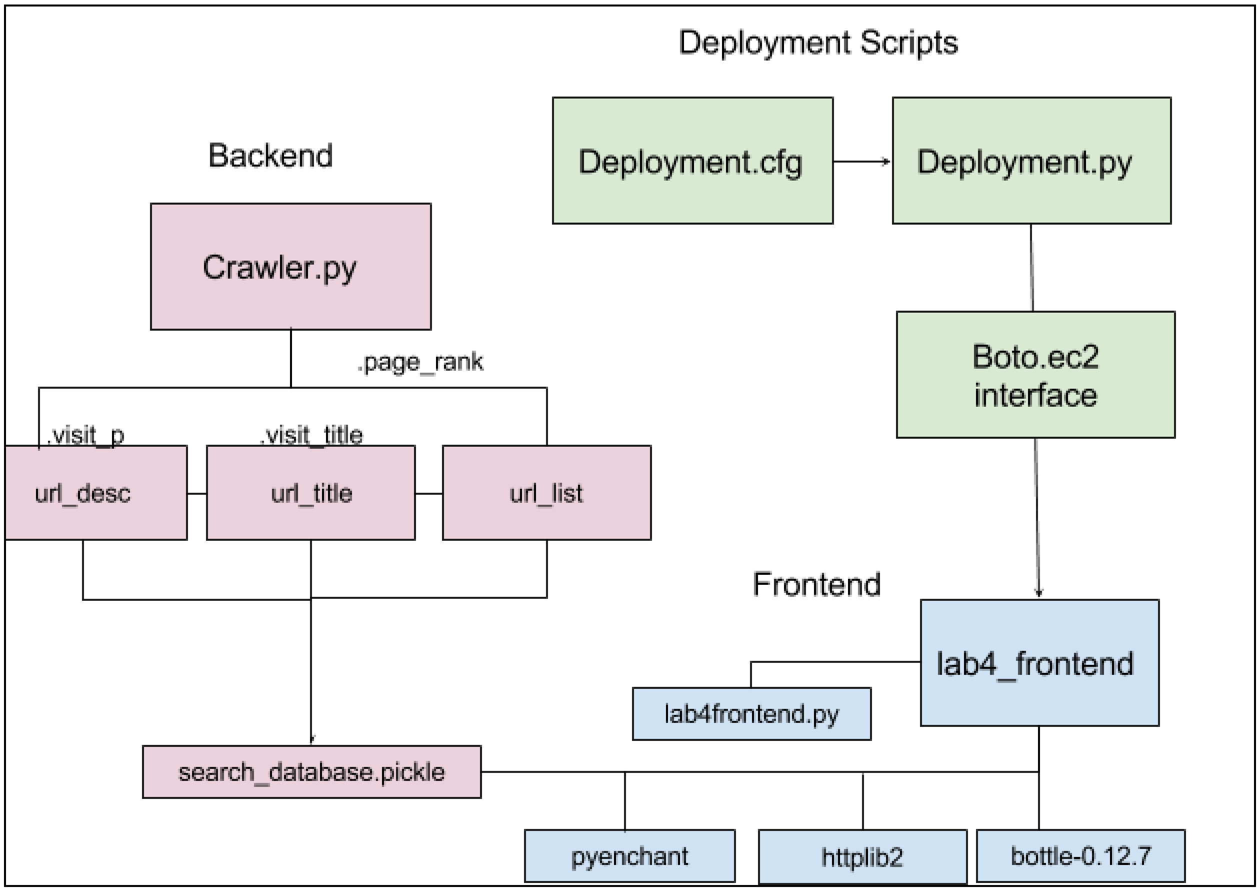
Computer Engineering Research Group
http://www.eecg.toronto.edu/Welcome.html
The department is divided for the purposes of graduate-level research activity into eight groups. The Computer Engineering Group was established arou...

7. **Backend - Obtain Description**
   The backend crawler code was enhanced to obtain the page description by searching through the different tags (i.e. <p> tags). Refer to the figure 5 above.

3) **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).**

All frontend code is located in lab4frontend.py. The css, html and python code for the frontend is included within this file. Note The css is under the <style> tags. Bottle is used as the frontend framework, as such all bottle files are kept inside the 'bottle-0.12-7' folder. For the backend code, refer to the UML Diagram below.

Figure 6: High Level UML Diagram



4) **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.**

Initially in our proposed design, we intended to use redis server, however in the final completed design we used pickling. This is due to the restriction that we had to make everything work locally.

5) **Explain your testing strategy during the development. Describe how you identify the corner cases.**
For both the frontend and backend, unit tests were created for smaller units and integration tests were created to confirm functionality of the entire search engine. This would ensure that no already working functionality is tampered with after new features are added. We also used the agile methodology for testing to iteratively test after every feature was added, this gave us confidence that we always had a working product.

Some of the front end corner cases that were considered include: adding extra spaces in the search bar, leaving the input empty and searching, searching repeated identical words and testing search suggest functionality by ensuring no suggestion is given when unmeaningful input is put in (i.e.

'lksjflkasdjflksjdflksjdfkj'). Each of these corner cases were identified by discussing and trying to poke holes at our architecture and assumptions we made while building the search engine.

6) **Lessons learned from this project.**
One of the biggest lessons learnt in the project was that website design and architecture should have been decided in prior labs. Since the design was constantly modified, a lot of work needed to be redone and repeated. Another important lesson learnt was that testing deployment scripts take time, therefore it is important to write them in advance.

7) **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?**

Things we would do differently if we had more time:
- Use a front end framework like react or angular.js
- Implement a multi-threaded crawler
- Be able to perform searches for i.e. similar to google images tab

Things we would do differently if we had to do it again:
- Implement the web design/UI portion in lab 1 this way we can focus adding more functionality for the final project

Tasks that took longer than expected include:
- Designing a responsive UI that adjusts to the devices
- Writing deployment and termination scripts - these scripts require a significant amount of time to run, therefore testing them as we wrote them was time consuming. Additionally, a lot of version errors were obtained while testing the scripts - this was not expected but a needed to be fixed problem nonetheless.

8) **How the material from the course helped you with the project.**
The material from the course supplemented the project - for example we learnt about decorators after the midterm which explained @route implementations within the project. A lot of the data structures such as dictionaries, and lists were also taught in class therefore building the project was more intuitive. Finally, learning about databases/storage in lecture introduced us to redis, sqlite etc which we were able to explore for the project.

9) **How much time it takes for you to complete each lab outside the lab sections.**
The amount of time it took to complete each lab varied by the lab:
- Lab 1: 20 hours
- Lab 2: 30 hours
- Lab 3: 25 hours
- Lab 4: 35 hours

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

We found the google api integration very interesting and useful. Managing sessions and implementing login can be easily extended to any project or even a simple mobile app. The labs could emphasize more on login functionality to allow us to explore building a login system without an external

Additionally, deploying on aws using scripts was also a very unique component of the project, it exposed us to how a company would be able to keep their application accessible. It also explains how multiple servers can be configured without needing to manually configure them.

Finally and most importantly learning the inner workings of web crawling and mining information was fundamental to our understanding of how a search engine works.

11) **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.**

We believe that benchmarking is the least useful for these labs. This is due to the fact that our search engine was modified frequently and therefore the results of benchmarking would also change frequently.

12) **Other feedback or recommendations for the course.**

We enjoyed the labs - they were fun and involved a lot of learning. We would be interested in having the freedom to explore more frameworks such as Django if python is continued to be taught.