Data Structure Proposal—Let’s Beat Google!

資管二	林彥彤	106306007
資管二	朱彬祺	106306008
資管二	洪浩晟	106306054
資管二	戴士翔	106306055
資管二	謝采辰	106306076

House renting

1. Motivation:

The information on the internet for house renting is too complicated, and by the manipulation from the websites, the information provided are often far from reality or contain too much advertising elements. Due to the biased algorithm from the house renting websites, many people rely on their family and friends to search for houses, thus not creating an efficient market for house renting because of enclosed market information.
So, we decided to create a search engine specifically for house renting, which not only improves the efficiency for searching but also satisfy the user's needs.

2. Function:

To use this searching engine, users can type the weight and keywords, then the system uses our formula to calculate a website's score, and then list out results from higher to lower scores. The system will only receive at most to three keywords, so that we can control the quantity of the variables.

3. Keyword and score formulation:

(1) Keyword:

We selected five keywords: 租屋/學生/傢俱/低價/車位/...etc.
The keywords are based upon the ones at Google in order to make sure that the results are related to house renting.

(2) Score formulation:

Score = weight * count

4. Algorithm:

(1) Score algorithm:

a. Setting weights for default keywords.
b. Retrieve the keyword data, calculate the word frequency based on the frequency of keywords on pages and subpages.
c. For word frequency statistics, assign weights according to word frequency, and multiply by weight to calculate score.
d. In order to return score, use our formula “Score = weight * count”
count defalt as 1. Weight is a default keyword[租屋, 學生, 傢俱, 低價, 車位]and the

weight is 1.5 , 1.4 , 1.3 ,1.2 ,1.1 in order.

Ex:
If the user enter a word that do not exist on the web, the formula will be:
score = weight * 1, the searching result will show 租屋、學生、傢俱……

Count is the weight converted from statsistical word frequency [3,2,1.5] after the user types a keyword.

Ex:
After entering a keyword, formula of computing will be:
score = weight * 3
Searching result will show the sorting list of keywords.

After entering two keywords, formula of computing will be:
score = weight * 3 + weight * 2

Every page’s total score is: the page itself’s score + subpage’s score.

Ex:
Web page A contains subpage B and C, the score for A web page is added by the score from b、c and itself.

(2) Sorting algorithm:

After computing all the scores, consider that the main page's score must be the highest, we:

a. Filter out all the pages with appearance of keyword more than three times.
b. Rank them from low to high.

The principle: We thought that the main page with the highest score can be easily found, but the message it contains is miscellaneous,too. Thus, we assume that the page with lowest score is the most detailed, and it is exactly what user want to found and should be ranked at the front. Filtering out pages with the appearance of keyword more than three times is setting up an standard and make sure this page do have the relationship with the keyword. After all, we can confirm that the page we get is related to our keyword and we won't get the main page, which contains the most information.

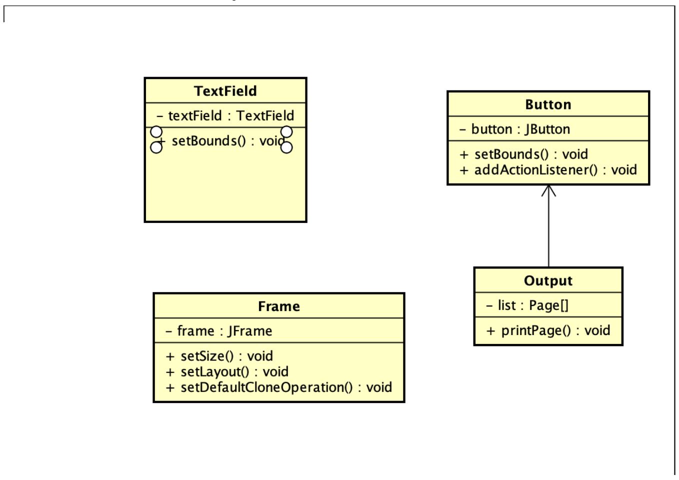
5. System design and class diagram

(1) Workflow:

First, the system will receive the weight as well as the keyword from the user. Then, through our formula, we can calculate the total score of a website. In the end, according to the score, list the websites from higher score to the lower ones.

(2) System structure:

Front-end Development: Provide the interface for users and export the results.

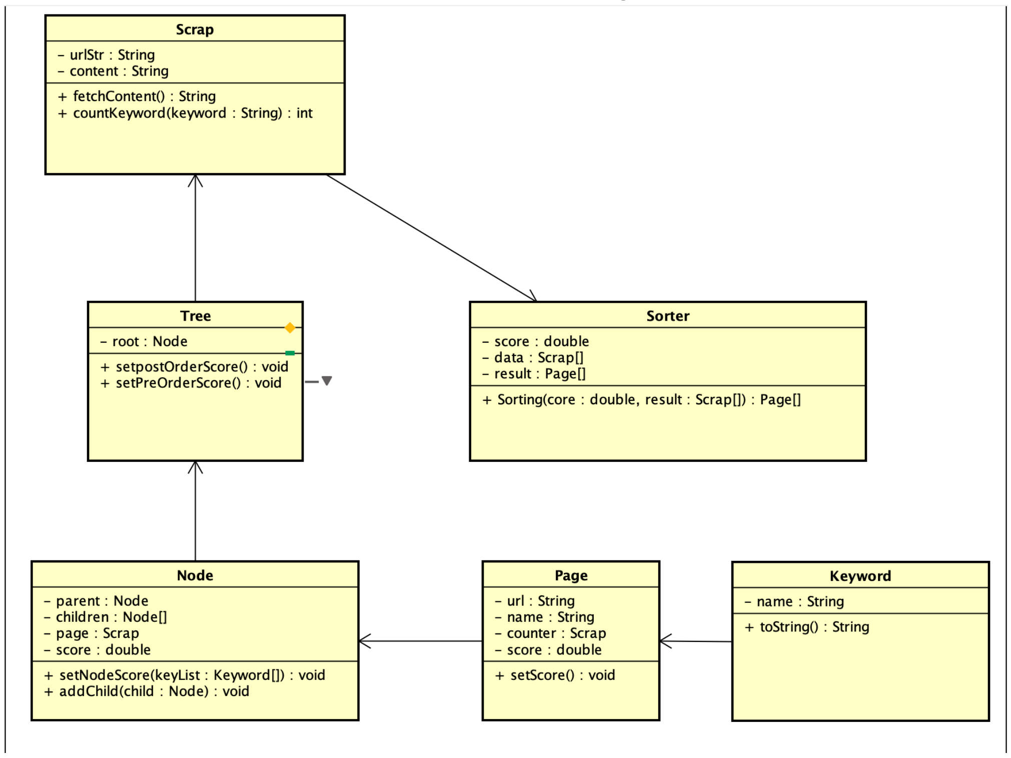


a. Receive the user's input, export to back-end
b. GUI and beautification
c. Export the result

Class used:

Textfield: Text description
Button: The mechainism to trigger an event
Frame: Setting the user interface
Output: Display the results

Back-end Development: Calculation and ranking



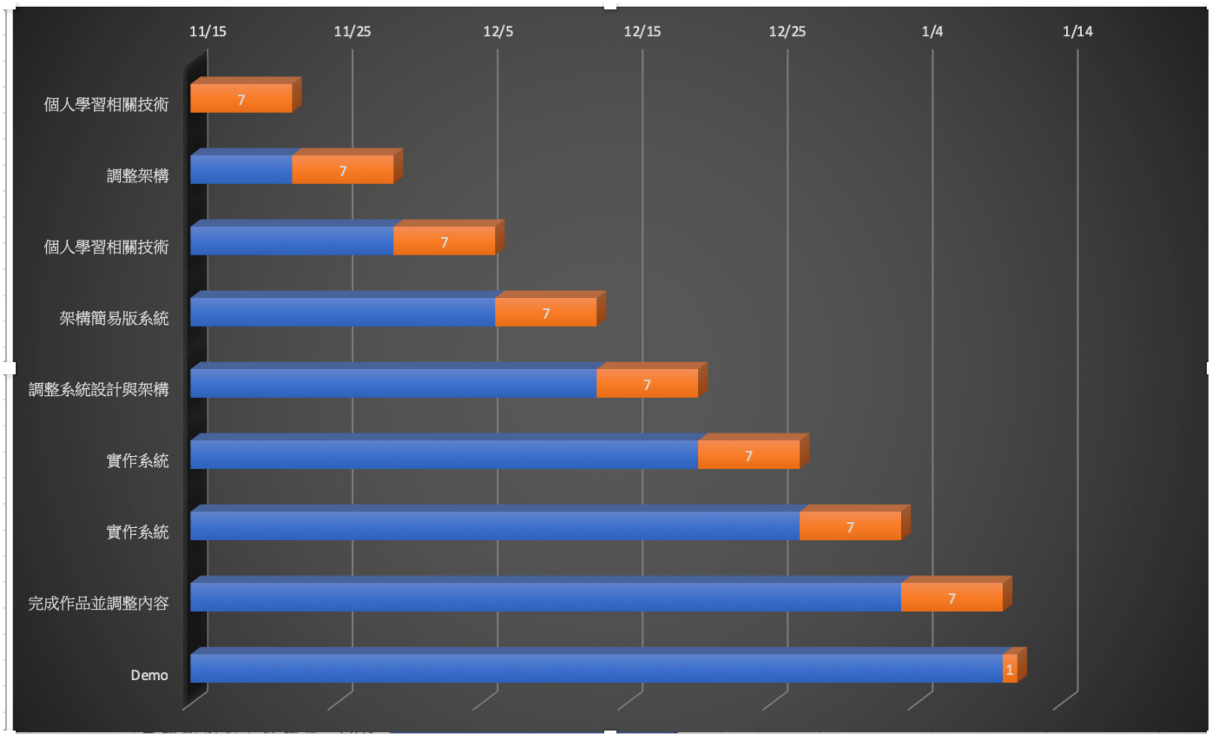
a. Receive the keyword and weight from front-end, use web-crawler to get information.
b. Sort the data and calculate the number of keywords.
c. Use formula to calculate the score of the page.
d. Connect subpage and do loop.

Class used:

Tree: Use main page as root, use subpages as trees of leaves
Node: Take each page as tha data to operate webpages.
Keyword: Establish keywords’ categories according to what front-end returns.
Sorter:Sort the score of pages based on sorting algorithm
Page: It represents the webpage that is stored, url means the location of the pages.
Scrapy: Crawl the data from webpages, use it as the matirials for comparing to keywords.

6. Schedules

11/22: Learn the techniques related to topic
11/29: Learn the techniques related to topic
12/6: Construct a simple version of the system
12/13: Adjust the system design
12/20: Run the system
12/27: Run the system
1/3: Finish the project
1/10: Demo



(1) Developing model:

We adopt Agile-development to develop the project. It is a human-centred, iterative, and step by step developing way. Accordingly, we set up a designing plan including system adjustment. We will test the system in phase 1. After confirming functions working properly, we will start expanding the original version. Also, we will extend the practical edition as well as keep adaptive planning for the period of product.

(2) Distribution:

We arrange two of our group members to program at the front and back end respectively, to meet the business need of Agile-development. After finishing the code of both the front end and back end, we will integrate them and frequent releases the product intensively one week before the demo date.

7. Challenge and techniques

Designing website GUI
Excess subpage links
Add searching suggestions (similar to Google’s function)