The Hong Kong University of Science and Technology

Comp 4321 Team 19 Project

A Web Search Engine

Report (Phase 2)

Team members:

Name: Wing San Luk Name: Lee Wasin

ITSC: wslukaa ITSC: wleead

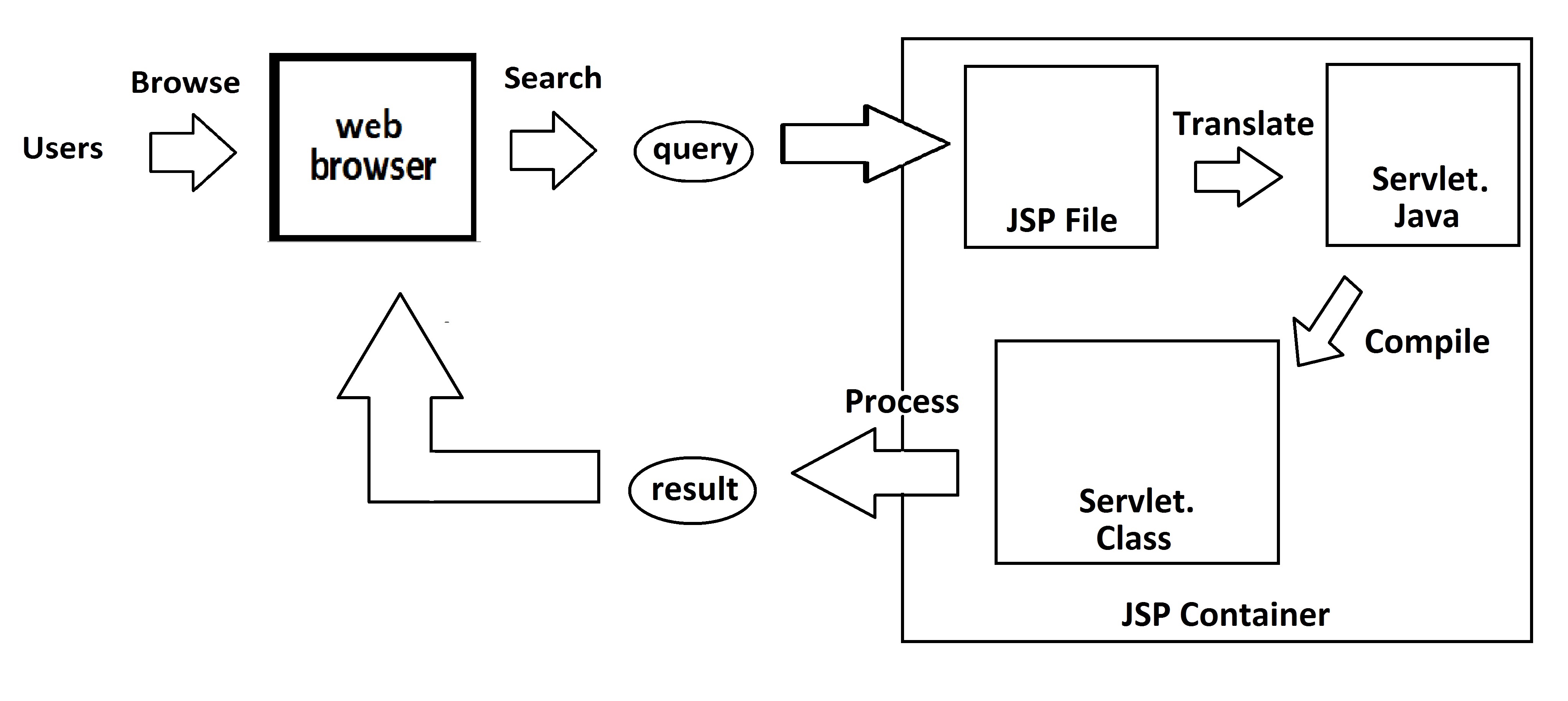
SID: 20193803 SID: 20266284

Name: Gao Xiao Wen Name: Cheung Wai Kwan

ITSC: xgaoah ITSC: wkcheungai

SID: 20264676 SID: 20272910

# 1. The Overall Design of the System



## 1.1 Flow of our Search Engine

Users will first go to our **web browser** and use our **search functions**. Then after they select the **keywords** and hit the search button, the query containing the search words will be sent to **JSP file**. A **servlet java** will be generated and the **container** will compile it. After the **class** is out, it will be executed and **produces the search results** which will be shown on our website.

## 1.2 Term Weight formula

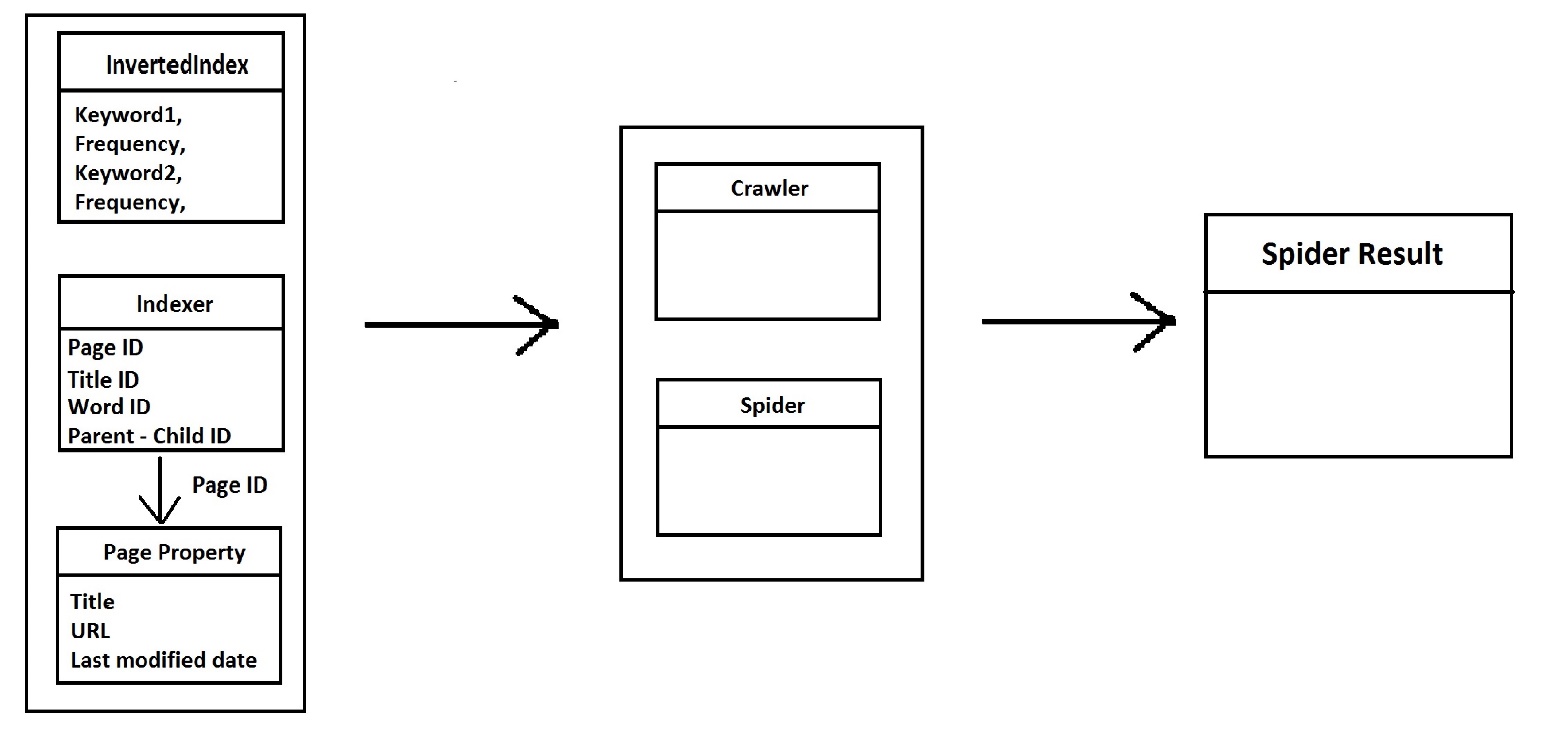
*wij =* (*tfij /maxl*{*tflj*}) *• idfj*

## 1.3 Cosine Similarity Measures



Query weight is set to 1

# 2. The File Structure of the indexed Database



Each class contains a hashtable which is used to store data and we can call the data by using the function “hashtable.get(index)”.

## 2.1 IndexerHelper class

Page **Indexer**

URL <-> PageID

Title Indexer

Title <-> TitleID

Word Indexer

Word <-> WordID

Child Parent

Parent-ID <-> ChildID

## 2.2 InvertedIndex class

Inverted index

word ID -> {pageID, Freq}

Forward index

Page-ID -> {keywords}

Link-based index

Parent-ID -> {ChildID}

## 2.3 InformationOfPage class

Page properties

Page-ID -> title, URL, last-date-of-modification, size

## 2.4 Explanation

After crawling a website, we will have a page ID from the current website. First, we can save Title, URL, Last-date-of-modification, Size and the Child page in the Index class because we need the index and basic information about the page.

Then, we will save the related Keywords and its Page ID and the Frequency it appeared in the website from the Inverted Index class. We chose to use Inverted index is because we can have a faster retrieval when we want to call the data.

# 3. Highlight of features beyond the required specification

## 3.1 Similar pages search

After searching the input work, we count top 5 maximum number of key words frequency when fetching a page and show it in the result page. Next to the that, there will be a “similar pages” button. By clicking the “get similar pages” button, top 5 most frequent keywords from the page will be used to search for similar pages. It allows users to have a higher chance of searching the page that they needed as it will neglect the “less related” keyword.

## 3.2 Lucky search

This button allows users to see 10 words from all the keywords indexed in our database. Users could choose to select the one they think is interesting and search it. It is suitable when the users do not know what to start with or the actual spelling of the word.

## 3.3 User Interface

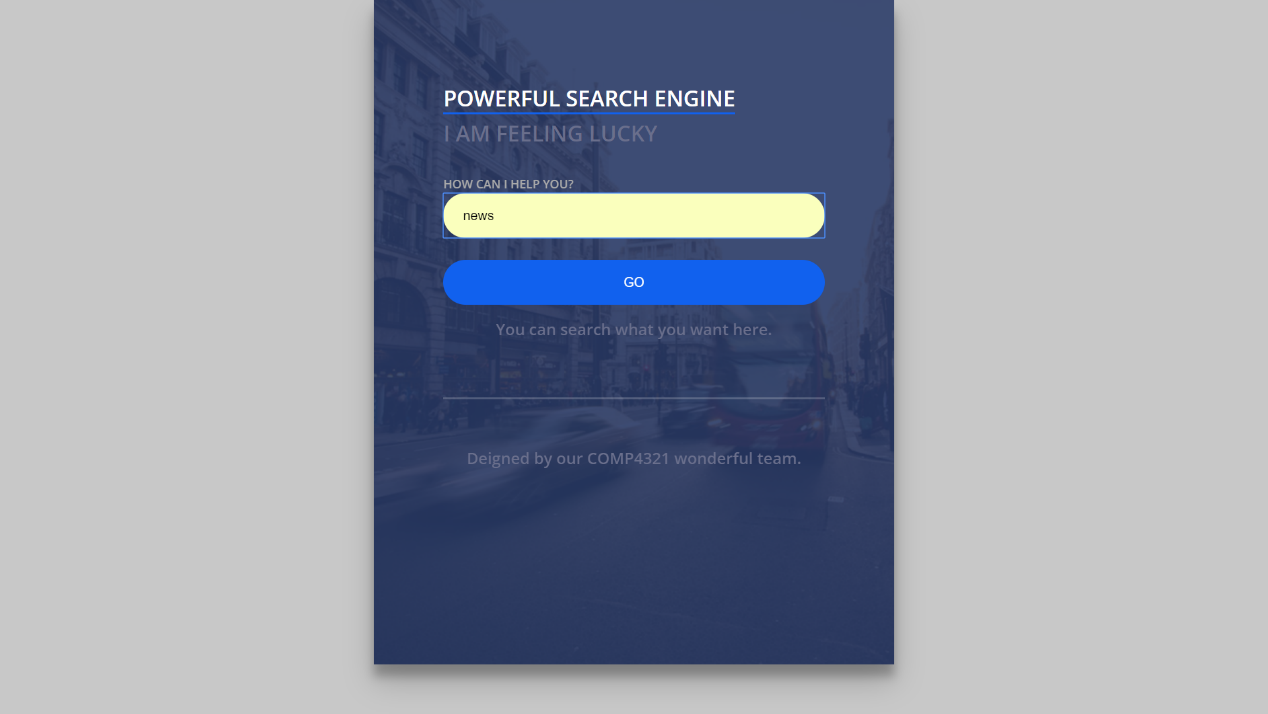
A user-friendly interface with clear linked buttons for all of the functions are created. It allows users to use the search engine more easily.

## 3.4 Link In Result Ranking

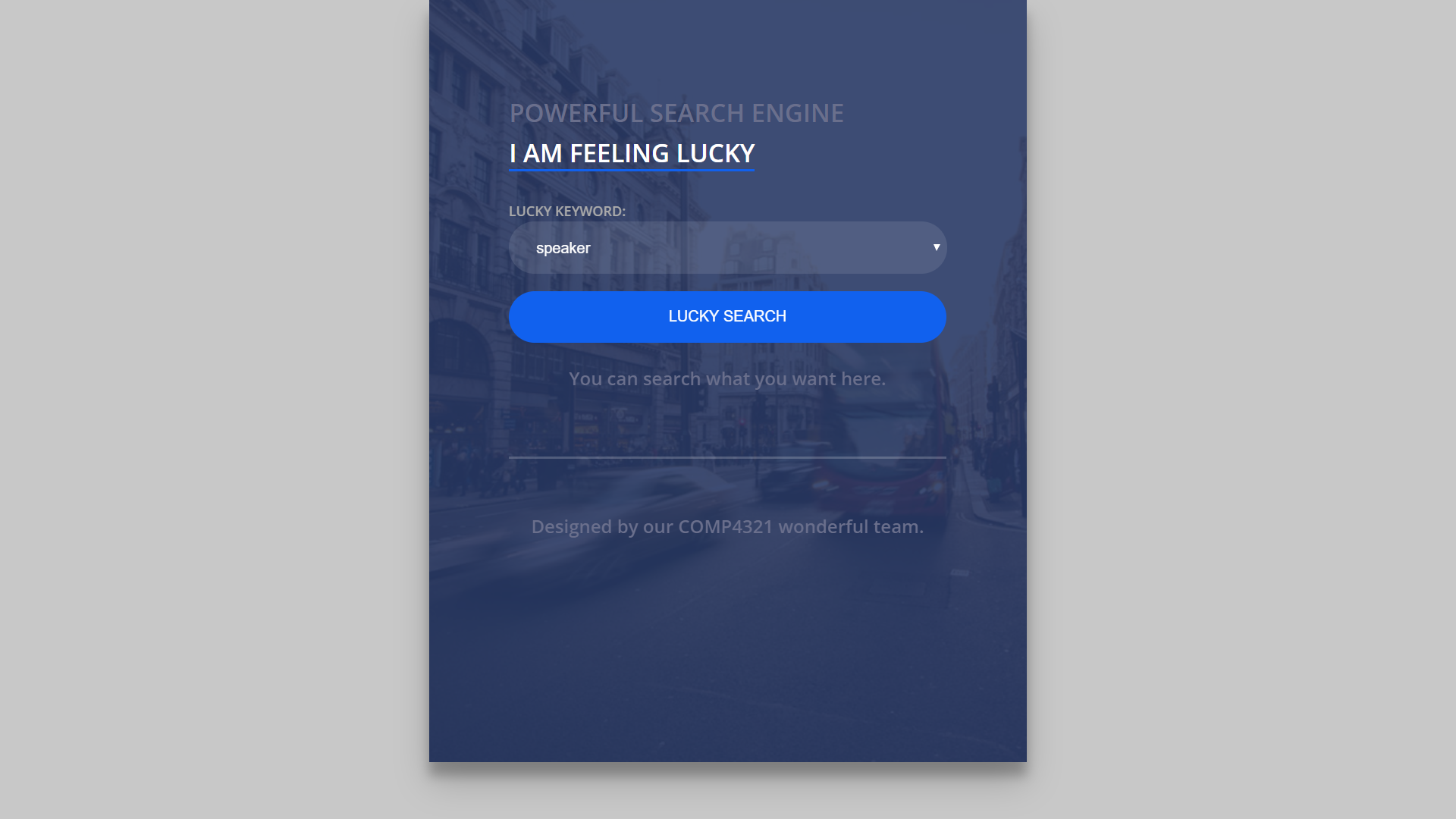
we add weight to the total score of a link if there are more parent links pointing to it. (Score of Link In Result Ranking).

# 4. Testing of the function implemented specification

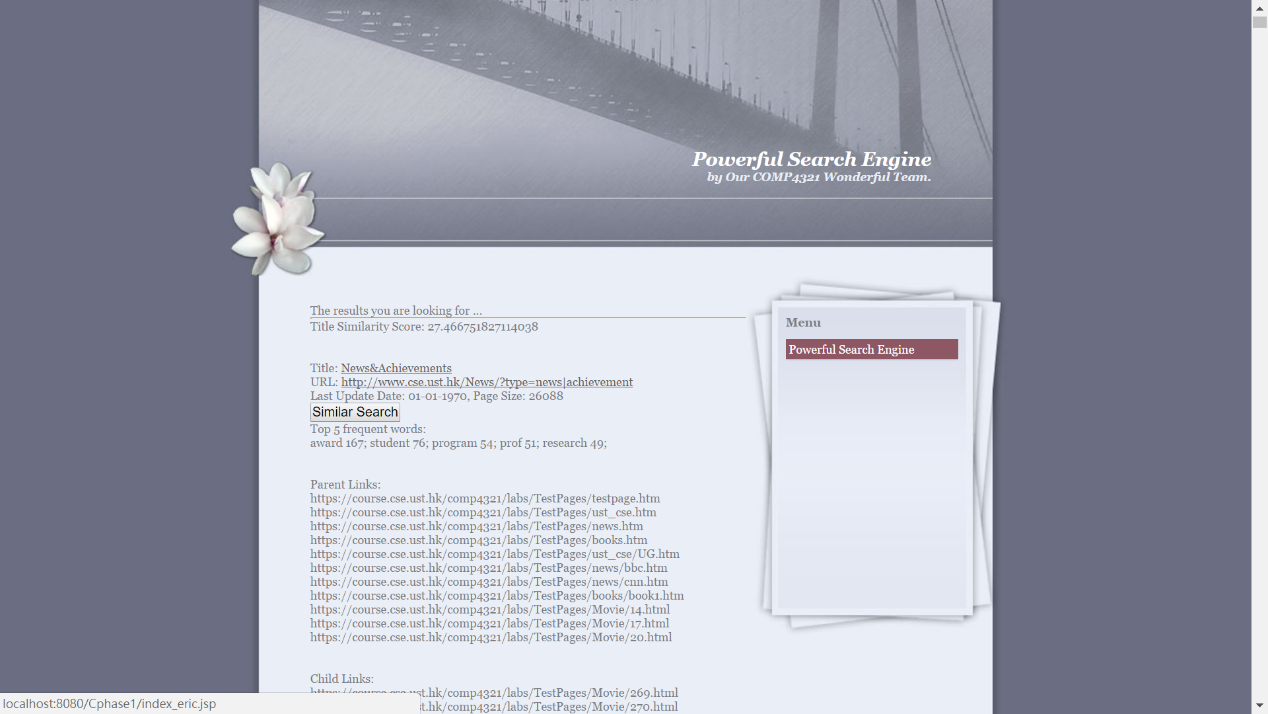
http://localhost:8080/Cphase1/index\_eric.jsp



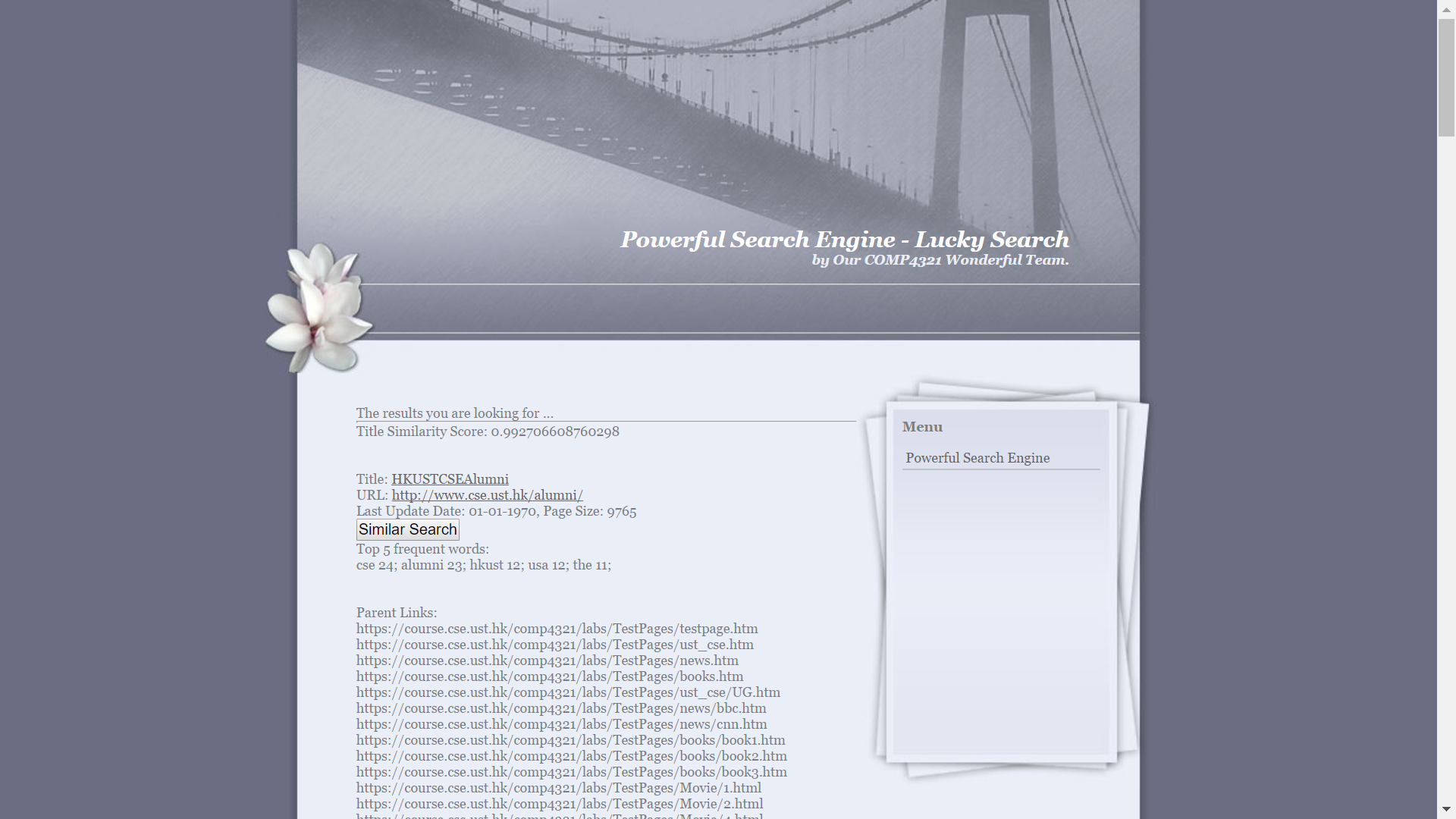
Lucky Search



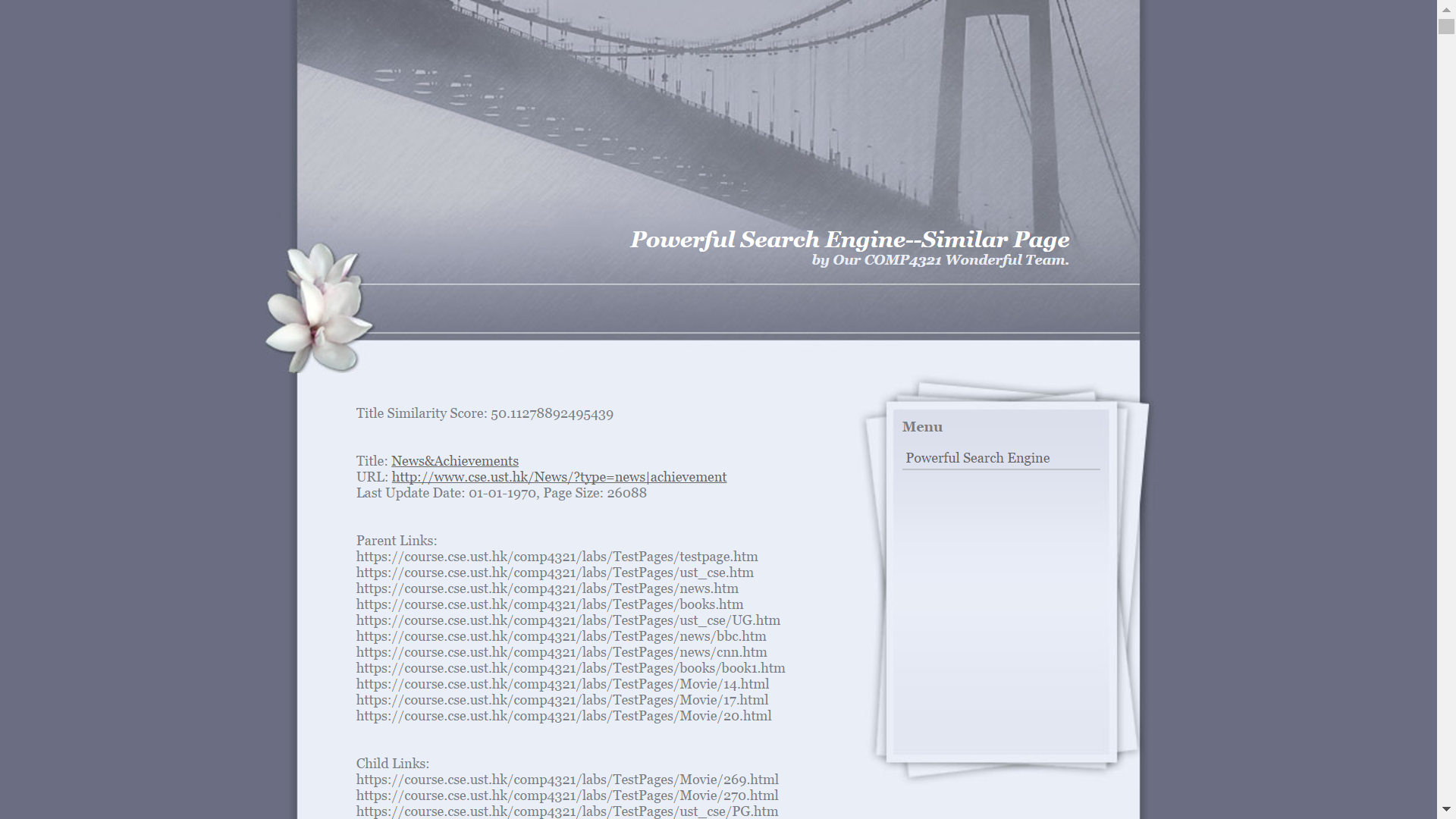
Powerful Search Results



Lucky Search Results



Similar Page Results



# 5. Conclusion

## 5.1 Strength

Use string to save the link

When we process the url, we pass it as a string which allows the search engine to run slightly faster.

Provide numerous options to search

We provide functions like “similar page search” and “lucky keyword search” to allow users to have a broader option to search and improve the chances of getting the wanted results.

## 5.2 Weakness

Compile after the first access

JSP can only be compiled when first accessed. Therefore, when we first compiled the JSP page, there will be a noticeable delay.

## 5.3 If time travels back…

Redesign the project

We should have divided the functions better. Due to time limit, we did not do well in refactoring. We should use more time on the design of the system.

## 5.4 Interesting features to be added

Suggested keyword when typing

Just like google and other search engines, when the users are typing their keywords, the suggested keywords will appear so to save users’ times.

Operators + short-forms search

We could include the search of some operators like &, ^ and short-forms like u, omg in order to be more user-friendly.

## 5.5 Percentage contribution and tasks performed by each project member

|  |  |  |
| --- | --- | --- |
| Name | Percentage contribution | Tasks |
| Wing San Luk | 25% | SearchEngine.java |
| Lee Wasin | 25% | Report(mainly), wasin.jsp (lucky search) |
| Gao Xiao Wen | 25% | Report, Katrina.jsp(similar page) |
| Cheung Wai Kwan | 25% | Report(final proof read), index\_eric.jsp (index page), winson.jsp (Output of results), UI design, Web-interface |