1. BFS Searching
    1. Get the Basic Page Info
        1. Last Modification Date (If missing, consider date field) (Testing Required)
        2. Size of Page (If missing, consider the total number of characters)
    2. Get the list of URLs and filter out the following URLs:
        1. URLs visited in the current iteration
        2. URLs in the index table that are not updated (i.e. last modification date in this website <= last modification in the index table)
    3. Words Extractor
        1. n-gram (1, 2, 3) to extract single or phrasal words
        2. extract heading, body respectively

2. Indexer
    1. Open the JDMB Manager and the related tables in it.
    2. In each page iteration:
        1. Remove stop words
        2. Store the original words
        3. Stem the words
        4. Find the words ID in the WordMapping
            1. If exists, use the word ID in the remaining steps
            2. If not exists, add a new mapping in the WordMapping
        5. Store the max(tf) in each document
    3. Calculate the tf-idf/max(tf)

Database Design Update:

- PageChildMapping
    - *Parent Page ID
    - [Children Page ID]

3. Search Engine
    1. Receive the list of word and use the same Word Extractor (From the project document, it said that phrases are specified in the query: “Hong Kong”)
    2. Find the similarity on tf-idf/max(tf) using cosine similarity
        1. Include mechanism to favour match in title (weighted similarity?)
    3. Return the top-50 results to the web interface

4. Web Interface
    1. Text Box to submit the query to the search engine (Need to clean the input?)

2. Output:

Each item returned is displayed in the following format:

score	page title
	url
	last modification date, size of page
	keyword 1 freq 1; keyword 2 freq 2; . . .
	Parent link 1
	Parent link 2
	... ...
	Child link 1
	Child link 2
	... ...
score	page title
	... ...

- The title and URL are hyperlinked to the actual page on the remote server
- The list of keywords displays up to 5 most frequent stemmed keywords (excluding stop words) in the page together with their occurrence frequencies

3. Get Similar Pages Button (Bonus)
    1. Extract the top 5 most frequent keywords (stem) and resubmit the query

4. Select the keywords from indexed word table (Bonus)

5. User-friendly (e.g. DHTML, AJAX, application-based interface)

4. Keep track of query history (need application-based interface or cookies, etc.) and allow users to view the result of a previous query (or operate on the results, e.g., merging the results of two previous queries or search within the result of a previous query).
5. Many other features that you can observe from existing commercial search engines.
6. Consider links in result ranking (e.g., PageRank).
7. Exceedingly good speed by using special implementation techniques.