Comets Engine: Cyberminer Search Engine

Software Architecture & Prototype Implementation

Team Members:

Shanjida Khatun (sxk200130@utdallas.edu)
Miao Miao (mxm190091@utdallas.edu)
Mohit Anand (mohit.anand@utdallas.edu)
Zilong Wang (zxw200004@utdallas.edu)
Jeya Visshwak Jeyakumar (jxj190055@utdallas.edu)
Mitha Alshammary (mi.alshammary@utdallas.edu)

Team Website:

https://personal.utdallas.edu/~zxw200004/SA/SE6362_21F.html

For SE6362.001

Professor: Dr. Lawrence Chung

Table of Contents

I. Overview

The goal of our project is to develop the KWIC index system (Key Word in Context) proposed by David Parnas in the early ’70s using Java Applet. This system provides a convenientsearch mechanism for information in a long list of lines, such as book titles, or online documentation entries.

In Phase 1 of this project, the KWIC system is designed, implemented, and tested to satisfy the functional and non-functional requirements. An architectural overview is provided of the KWIC system. Based on the design, the system is implemented using Java. We have described a user manual as a guideline.

In Phase 2, we have built on top of the phase one KWIC system to create a web search engine called Cyberminer. Implementing this phase required the use of an Object-Oriented architectural style and building a Java applet or a similar application. Cyberminer is hosted on the team’s webpage. The program shall satisfy both functional and non-functional requirements listed below by implementing different features.

II. The Cyberminer Search Engine System

1. Functional Requirements

Cyberminer shall accept a list of keywords and return a list of URLs whose descriptions contain any of the given keywords. Cyberminer shall use another software system, the KWIC system, as a component, to efficiently maintain a database of URLs and the corresponding descriptions.

The KWIC (Key Word in Context) index system shall accept an ordered set of lines, where each line is an ordered set of words, and each word is an ordered set of characters. Any line shall be “circularly shifted” by repeatedly removing the first word and appending it at the end of the line. The KWIC index system shall output a listing of all circular shifts of all lines in ascending alphabetical order, together with their corresponding URLs. No line in the output list shall start with any noise word such as “a”, “the”, and “on”.

The KWIC system shall allow two modes of operation:

i) for building initial KWIC indices.
ii) for growing the indices with later additions.

Cyberminer shall allow the following operations:

i. Case sensitive search
ii. Hyperlink enforcement
iii. OR/AND/NOT Search

iv. Multiple search engines
v. Deletion of out-of-date URL
vi. Output is displayed in ascending alphabetical order
vii. Setting the number of results to show per page and navigation between pages
viii. Autofill while correcting typographical errors
ix. Filtering out symbols that are not meaningful according to the user configuration

Issues:

FR-1: Cyberminer shall accept a list of keywords to search and return a list of URLs whose descriptions contain any of the given keywords.

Issue Description: The requirement is ambiguous as it does not clearly define how the input is provided to the system.

Option 1: The input can be entered from the keyboard manually in the given input field.
Option 2: The input can be read from the file selected.
Option 3: The input can be copy-pasted into the given input field.

Decision and Rationale: Option 1 and 3
The input is provided to the system by the users either by entering a set of lines in the input field manually through a keyboard or by copy-pasting. No other input ways will be permitted.

FR-2: Cyberminer shall accept a list of keywords to search and return a list of URLs whose descriptions contain any of the given keywords.

Issue Description: The requirement is ambiguous as it does not clearly define if the description is also returned along with the URL in the search result.

Option 1: The description is shown along with the URL in the search result.
Option 2: Only URL is displayed in the search result.

Decision and Rationale: Option 1
For our Cyberminer System, we will be considering displaying both URL and description in the search result. This will provide a better contextual search experience and increase the probability of choosing the desired URL.

FR-3: Cyberminer shall use another software system, the KWIC system, as a component, to efficiently maintain a database of URLs and the corresponding descriptions.

Issue Description: The requirement is ambiguous as it does not clearly define whether Cyberminer implements the KWIC system functionality or will have a KWIC system as a component.

Option 1: The Cyberminer will implement the functionality of the KWIC system.
Option 2: The Cyberminer will have the KWIC system as a component.

Decision and Rationale: Option 2
For our Cyberminer System, we will use the KWIC system as a component. This provides us flexibility to modify the KWIC system or indexing algorithms in the future.

FR-4: The KWIC system shall accept an ordered set of lines, where each line consists of two parts:

i. The URL part, whose syntax is in the spirit of the following:
   URL ::= ‘http://’ identifier ‘.’ Identifier ‘.’ [‘edu’ | ‘com’ | ‘org’ | ‘net’]
   identifier ::= {letter | digit}+
   letter ::= [‘a’ | ‘b’ | … | ‘y’ | ‘z’ | ‘A’ | ‘B’ | … | ‘Y’ | ‘Z’]
   digit ::= [‘1’ | ‘2’ | … | ’9’ | ‘0’]
ii. The descriptor part, whose syntax is:
   identifier {‘ ‘ identifier}*

Issue Description: No issues noticed. The requirement is concise and complete.

FR-5: The descriptor part of any line shall be “circularly shifted” by repeatedly removing the first word and appending it at the end of the line.

Issue Description: The requirement is ambiguous as it does not clearly define if we need to consider all the lines to circularly shift or only a few.

Option 1: All the lines should be circularly shifted.
Option 2: Only a few lines should be circularly shifted.

Decision and Rationale: Option 1
All the lines should be circularly shifted or else the system would generate an incomplete set of shifted lines, which would lead to ineffective search results and decrease the efficiency of the KWIC Search Engine.

FR-6: The KWIC index system shall output a listing of all circular shifts of the descriptor parts of all lines in ascending alphabetical order, together with their corresponding URLs.

Issue Description: The requirement is ambiguous as it does not clearly define if the output display all the circularly shifted lines from the KWIC system along with the URL.

Option 1: Display the array of circular shifted lines of the descriptor along with the URL in the search result.
Option 2: Display only the original descriptor part stored along with the URL in the search result.

Decision and Rationale: Option 2
To make the Cyberminer system more user-friendly, we will only display the original descriptor part stored along with the URL in the search result. This provides abstraction and hides the internal processing aspects of the end-user.

FR-7: No line in the output list shall start with any noise word such as “a”, “the”, and “on”.

Issue Description: The requirement is ambiguous as it does not clearly define if the noise words are limited to only “a”, “the”, and “on”.

Option 1: More noise words can be identified and added to the noise eliminator component of the KWIC system.
Option 2: “a”, “the”, and “on” are the only noise words.

Decision and Rationale: Option 1
Including more noise words would reduce the words to be processed thus lowering the processing time and increasing the performance of the system.

FR-8: The KWIC system shall allow for two modes of operation:
i) for building an initial KWIC indices, and
ii) for growing the indices with later additions.

Issue Description: No issues noticed. The requirement is concise and complete.

FR-9: The system shall store the input as given and retrieve the input also as such case sensitive search.

Issue Description: No issues noticed. The requirement is concise and complete.

FR-10: When the user clicks on the URL, which has been retrieved as the result of a query, the system shall take the user to the corresponding website.

Issue Description: The requirement is incomplete as it does not clearly define if we need to
open the URL in the same tab or new tab or new window.

Option 1: The system will open the URL in the same tab.
Option 2: The system will open the URL in the new tab.
Option 3: The system will open the URL in the new window.

Decision and Rationale: Option 2
To use the Cyberminer for the next search it would be best to open the URL in the new tab. This would enhance the user experience.

FR-11: Specifying OR/AND/NOT Search: A keyword-based search is usually an OR search, i.e., a search on any of the keywords given. The system shall allow the user to specify the mode of search, using “OR”, “AND” or “NOT”.

Issue Description: The requirement is ambiguous as it does not clearly define how the system will identify the “OR”, “AND” or “NOT” mode of search.

Option 1: Provide a dropdown on the user interface to allow the user to select the mode of the search operation.
Option 2: Symbols ‘&&’, ‘!’ and ‘ ‘ shall be used for “AND”, “NOT” and “OR” search, respectively.

Decision and Rationale: Option 2
Including these symbols will be more user-friendly as most people are accustomed to Google Search and would find it easy to adapt and search without difficulty. Moreover, adding a dropdown will not be intuitive for the search query.

FR-12: Cyberminer shall allow for multiple search engines to run concurrently.

Issue Description: No issues noticed. The requirement is concise and complete.

FR-13: Cyberminer shall allow the deletion of out-of-date URLs and corresponding descriptions from the database.

Issue Description: The requirement is ambiguous as it does not clearly define if the out-of-date URLs should be deleted automatically or manually by the user.

Option 1: The user should be able to delete a specific entry from the database.
Option 2: Invalid and non-existent URLs should be deleted from the database automatically after a specific time based on a criterion.

Decision and Rationale: Option 1
For the Cyberminer system, the user is considered as system admin who maintains the database and is responsible for maintaining its consistency. Thus, the user will be able to remove the out-of-date URLs and their corresponding descriptions manually from the database.

FR-14: Cyberminer shall allow listing of the query result in ascending alphabetical order, most frequently accessed order, or per payment.

Issue Description: The requirement is ambiguous as it does not clearly define if we need to implement one or all the sorting features mentioned.

Option 1: Sort the query result based on the URL.
Option 2: Sort the result based on the URL's description.
Option 3: Sort the result based on the URL hit rate.

Decision and Rationale: Option 2
Implementing Option 2 for the Cyberminer system will display the top of the relevant website of the search result.

FR-15: Cyberminer shall allow for setting the number of results to show per page.

Issue Description: No issues noticed. The requirement is concise and complete.

FR-16: Cyberminer shall allow easy navigation between pages of the search results.

Issue Description: No issues noticed. The requirement is concise and complete.

FR-17: Cyberminer shall allow for Autofill while correcting typographical errors.

Issue Description: The requirement is incomplete as it does not clearly explain if the autofill happens for the entire sentence.

Option 1: Autofill only the words which are present in the database.
Option 2: Autofill the entire sentence by showing the descriptions from the database.

Decision and Rationale: Option 1
Displaying the whole sentence will not be considered a keyword search. Hence tokenizing the sentence into keywords and autofill the input search box based on the keywords will be the best solution.

FR-18: Cyberminer shall allow for Filtering out symbols that are not meaningful, according to the user configuration.

Issue Description: The requirement is incomplete as it does not mention when this filtering should have to be done.

Option 1: Filter out absurd symbols during the insertion of URL and descriptor.
Option 2: Filter out absurd symbols during the search result.

Decision and Rationale: Option 1
Filtering out absurd symbols during the insertion of URLs will reduce the complexity of the system.

2. Non-Functional Requirements

The Cyberminer system shall be easily understandable, user-friendly, portable, responsive, enhanceable, and reusable with good performance. The Cyberminer system shall also be adaptable.

Issues:

NFR-1: The Cyberminer system is easily understandable.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “understandable”.

Option 1: The system interface shall contain labeled buttons to make the application understandable to its users.
Option 2: The system shall have a properly defined method of instruction to guide the users on how to use the application.
Option 3: The system shall have a name or small description which briefly tells the users about the system.

Decision and Rationale: Option 1, 2 and 3
To make the system more understandable the users must understand what the application is all about. The idea is that a user who is using this application for the very first time should have a good capability of understanding the system. The user should not face any confusion during the entire time they spend while using the application and therefore the application must also not be time-consuming for the user to work with.

NFR-2: The Cyberminer system is user-friendly.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “user-friendly”.

Option 1: The system shall provide the necessary outputs desired by the user by taking in minimum input from the user.
Option 2: The GUI elements shall have labels that explain their function.
Option 3: The system shall have a help option that should have instructions on how to use the application.
Option 4: The system shall have a popup menu.
Option 5: The system shall be available in multiple languages.

Decision and Rationale: Option 1, 2 and 3
Users always prefer to provide a minimum number of inputs to get their desired outputs. Therefore, giving the minimum number of clicks on the user part would make the system user friendly. The GUI labels will help average users understand the functionality of each element very easily. The help option will help guide the user on how to use the system without needing the help of anyone. However, a popup may increase confusion among the users and might affect the user-friendliness of the system. Developing the system in

multiple languages can greatly increase both the implementation time and the cost of the project.

NFR-3: The Cyberminer system is portable.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “portable”.

Option 1: The system shall be accessible from all web browsers.
Option 2: The system shall be designed to be executable on any PC with Java VM installed.

Decision and Rationale: Option 1 and Option 2
Portable means the ease of use of this application from various platforms and environments. Therefore, any device that can connect to the internet and with a browser having a java plugin installed will have the capacity to use this application regardless of which operating system is being used.

NFR-4: The Cyberminer system is responsive.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “responsive”.

Option 1: The system shall give feedback to the users on both correct and incorrect actions.
Option 2: The system shall give the response as quickly as possible.

Decision and Rationale: Option 1 and 2
A response from the system is crucial to understand if the system obtained the actions of the user. It also helps the user understand whether they have provided the correct input for eg: prompting error message if wrong input is given. The system shall also provide this response as fast as possible so the user can respond to his previous actions in minimal time.

NFR-5: The Cyberminer system is enhanceable.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “enhanceable”.

Option 1: The system shall allow new features to be added if required.
Option 2: The system shall be easily updated and modified.

Decision and Rationale: Option 1 and 2
Enhanceable means that the system shall be updated if it does not meet some requirements or if the program contains some bugs. There shall always be options to modify a system to have a better lifespan and it also defines the quality of the software.

NFR-6: The Cyberminer system is reusable.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “reusable”.

Option 1: The code shall be well documented for future use.
Option 2: Parts of the system shall be used in the same style as the system was developed.
Option 3: The application shall be reused to develop the same application in different styles and applications.

Decision and Rationale: Option 1 and 2
Maintaining the documentation of the codes would decrease work and will also reduce confusion among new developers or while carrying out changes to the system. Using the part of the systems in the same style would help future developers. However, changing the styles to develop the same system would increase redundancy and duplicity.

NFR-7: The Cyberminer system has a good performance.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “good performance”.

Option 1: The system shall give a response in the shortest time possible.
Option 2: The system shall provide all the URLs associated with the given input.

Decision and Rationale: Option 1 and 2
Good performance means both good speed and correctness of a system. The system tries as much to not take users valuable time given that it also provides the user with all the desired outputs.

NFR-8: The Cyberminer system is adaptable.

Issue Description: The requirement is ambiguous as it does not clearly define what is meant by the word “adaptable”.

Option 1: The system shall adapt to any changes that go through in the system.
Option 2: The application shall work without an internet connection.

Decision and Rationale: Option 1
The system should adapt to any changes that get carried out during system updates which means if one module is changed the other modules will work as intended. However, the system cannot work without an internet connection as it is a web application.

III. The Deliverables

All the project materials are available online (website address given on the first page) and offline in One Drive (accessible to only group members of the team). The WRS-style format is chosen for documentation of the project in which the major sections typically include Overview, Functional Requirements, Non-Functional Requirements, Traceability Matrix, and Architecture Specification for the system. The presentation for the presentation slidesis made using PowerPoint and the slides are also available online.

1. Requirements Specification

a. Revised Functional Requirements

Cyberminer shall accept a list of keywords in the input field manually through the keyboard or by copy-pasting and return a list of URLs and their corresponding descriptions as a search result if those descriptions contain any of the given keywords. Cyberminer shall use another software system, the KWIC system, as a component, to efficiently maintain a database of URLs and the corresponding descriptions.

The KWIC system shall accept an ordered set of lines, where each line consists of two parts:
URL part and Descriptor part.

i. The URL part, whose syntax is in the spirit of the following:
URL = ‘http://’ identifier ‘.’ Identifier ‘.’ [‘edu’ | ‘com’ | ‘org’ | ‘net’]
identifier = {letter | digit}+
letter = [‘a’ | ‘b’ | … | ‘y’ | ‘z’ | ‘A’ | ‘B’ | … | ‘Y’ | ‘Z’]
digit = [‘1’ | ‘2’ | … | ’9’ | ‘0’]

ii. The descriptor part, whose syntax is:
identifier {‘ ‘ identifier}*

All the lines of the descriptor shall be “circularly shifted” by repeatedly removing the first word and appending it at the end of the line. The KWIC index system shall output a listing of all circular shifts of the descriptor parts of all lines in ascending alphabetical order, together with their corresponding URLs. No line in the output list shall start with any noise work such as “a”, “the”, and “on”.

The KWIC system shall allow for two modes of operation:

i) for building initial KWIC indices, and
ii) for growing the indices with later additions.

Cyberminer shall allow the following operations:

- Case sensitive search
- Hyperlink enforcement
- OR/AND/NOT ['&&', '!' and ' ' (space)] Search
- Multiple search engines
- Deletion of out-of-date URL
- Output is displayed in ascending alphabetical order
- Setting the number of results to show per page and navigation between pages
- Autofill while correcting typographical errors

b. Revised Non-Functional Requirements

The Cyberminer system shall be easily understandable such that even the first-time users will be able to understand the system and its usage. The Cyberminer system is developed using JavaScript with Object-Oriented Approach. This makes the system accessible on any web browser connected to the internet thus making it portable.

The Cyberminer system is enhanceable and allows updates on the system for bug-fixing and new feature inclusions. The Cyberminer system is reusable and can be used as a component if we are building any other applications which require search functionality. This would decrease the work and time required to build the system once again from the very beginning.

The Cyberminer system has good performance and displays the output on the screen within minimal time. Here the output is the alphabetically sorted descriptor lines along with corresponding URLs. The Cyberminer system is adaptable to the changes during system updates which mean if one module is changed the other modules will work as intended.

The Cyberminer system is user-friendly and requires only a minimum number of clicks to obtain the output. The help option on the screen also provides users with information on how to use the system and error troubleshooting. The Cyberminer system is responsive, and the GUI is also designed in such a way that users will be prompted with error messages on invalid inputs and displays output on valid inputs.

c. Requirements Traceability Matrix

	FR 1	FR 2	FR 3	FR 4	FR 5	FR 6	FR 7	FR 8	FR 9	FR 10	FR 11	FR 12	FR 13	FR 14	FR 15	FR 16	FR 17	FR 18
NFR1		✓		✓			✓		✓	✓	✓		✓		✓			✓
NFR2	✓	✓		✓	✓	✓			✓	✓	✓	✓		✓	✓		✓	✓
NFR3												✓						
NFR4	✓									✓		✓		✓		✓	✓	
NFR5								✓										
NFR6			✓				✓	✓										
NFR7						✓	✓		✓			✓				✓		✓
NFR8			✓				✓	✓					✓					

Fig. 1. FRs vs. NFRs Traceability Matrix Table

Functional Requirement (FR)	Architecture Specification (AS)
FR-1: The system accepts the input either by entering a set of lines in the input field manually through a keyboard or by copy-pasting.	**AS-1:** Cyberminer accepts ordered set of lines through insert module within the Cyberminer engine.
FR-2: The system displays both URL and description in the search result	**AS-2:** Cyberminer uses the print module inside the output function to display the URL and the description.
FR-3: The system will use the KWIC system as a component	**AS-3:** The system will have multiple tabs to allow users to use the KWIC system as well as Cyberminer.
FR-4: The system accepts an ordered set of lines, where each line consists of two parts.	**AS-4:** Cyberminer uses the print module inside the output function to display the URL and the description.
FR-5: The system shall construct circular shifts for all the input lines.	**AS-5:** Cyberminer provides a circular shift module to perform a circular shift on each input of an ordered set of lines
FR-6: The KWIC index system shall output a listing of all circular shifts of the descriptor parts of all lines in ascending alphabetical order, together with their corresponding URLs.	**AS-6:** KWIC System provides an output medium module that displays both the circular shifted lines and alphabetized circular shifted lines, together with their corresponding URLs.
FR-7: No line in the output list shall start with any noise word such as “a”, “the”, and “of”.	**AS-7:** The system uses a noise world filter function after circularly shifting the words for eliminating the noise words.

FR-8: The KWIC system shall allow for two modes of operation: i) for building initial KWIC indices, and ii) for growing the indices with later additions.	**AS-8:** KWIC system provides an option to update the system.
FR-9: The system shall store the input as given and retrieve the input also as such case sensitive search.	**AS-9:** Cyberminer provides an output module that prints and stores the inputs and outputs.
FR-10: When the user clicks on the URL, which has been retrieved as the result of a query, the system shall take the user to the corresponding website.	**AS-10:** Cyberminer guides the user to the website through a new tab.
FR-11: The system shall allow the user to specify the mode of search, using "OR", "AND" or "NOT".	**AS-11:** Cyberminer provides the user with a drop-down menu consisting of these options.
FR-12: Cyberminer shall allow for multiple search engines to run concurrently and allow access for any operating system in any device having an internet connection.	**AS-12:** Cyberminer uses multiple tabs to run multiple search engines separately.
FR-13: Cyberminer shall allow the deletion of out-of-date URLs and corresponding descriptions from the database.	**AS-13:** Cyberminer provides an option to update the system.
FR-14: Cyberminer shall allow listing of the query result in ascending alphabetical order, most frequently accessed order, or per payment.	**AS-14:** Cyberminer uses the query method inside the Cyberminer engine to satisfy this requirement.
FR-15: Cyberminer shall allow for setting the number of results to show per page.	**AS-15:** Cyberminer provides the user with a drop-down menu consisting of these options.
FR-16: Cyberminer shall allow easy navigation between pages of the search results.	**AS-16:** Cyberminer has labeled page numbers.
FR-17: Cyberminer shall allow for Autofill while correcting typographical errors.	**AS-17:** Cyberminer provides useful suggestions for spelling errors by using the English dictionary.
FR-18: Cyberminer shall allow for Filtering out symbols that are not meaningful, according to the user configuration.	**AS-18:** The system provides a noise world filter function that eliminates the non-meaningful words as noise words.

Fig. 2: FRs vs. ASs Traceability Matrix Table

2. Architectural Specification

a. Abstract Data Type for KWIC System

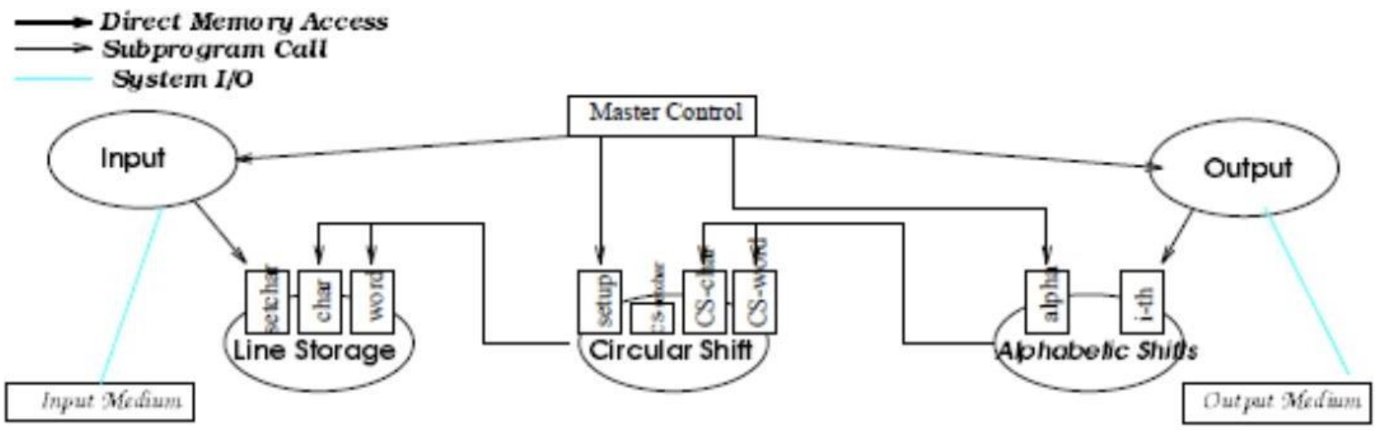


Fig. 3: Abstract Data Type For KWIC

In ADT architecture design, the components are objects. The system is divided into ADT objects,each handling a specific aspect of system functions. Each ADT object provides an interface to communicate with other objects. An object is responsible for preserving the integrity of its representation that is hidden from other objects. So, data is not directly shared by different objects,but through explicitly invoking interfaces.

b. Proposed Abstract Data Type for KWIC System

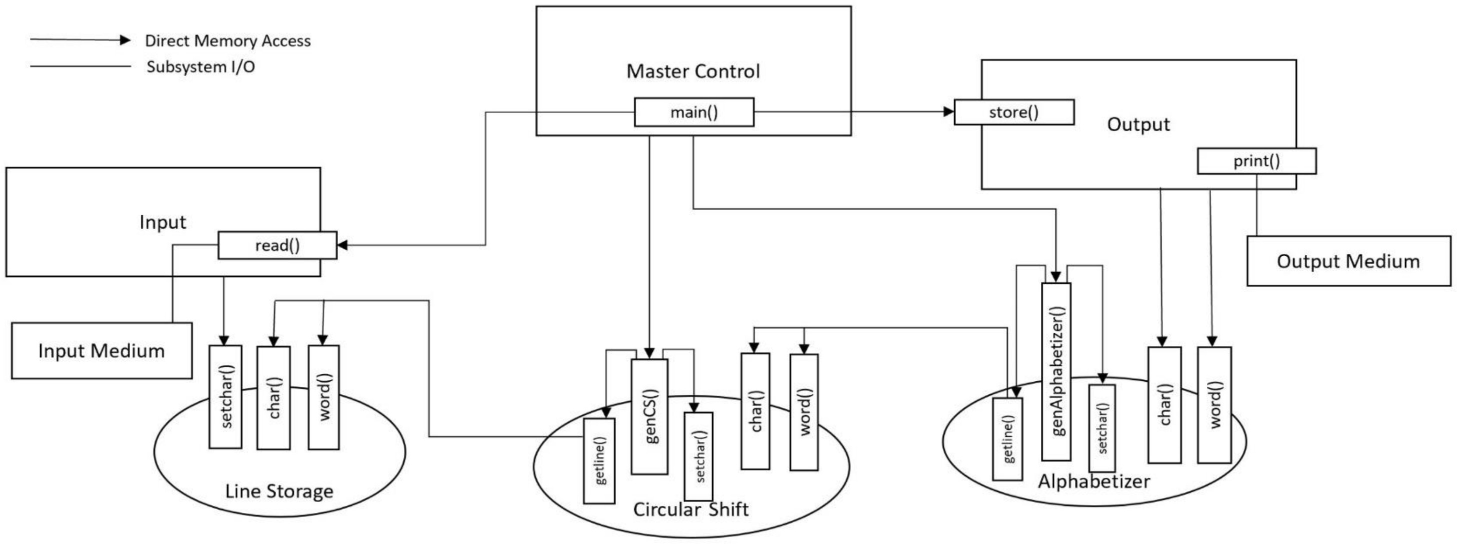


Fig. 4: Improved Abstract Data Type For KWIC

Components

Module Input: Master control passes control to input, then input reads data lines from the inputmedium and stores data line. Input invokes the “setchar()” method offered by the Line storage.

Module Line Storage: This module will create, access, and possibly delete characters, words, andlines. It provides interfaces “setchar()”, “word” and “char()”. Char and Word are then invoked by the “getLine()” interface of the module circular shift.

Module Circular Shift: Circular shift is invoked by master control through “genCS()” interface. The circular shift means repeatedly removing the first word and appending it at the end ofthe line until a circular is reached. It constructs a circular shift of the words generated by the “genCS()” interface. The “char()” and “word()” interfaces are then invoked by the next “getline()” interface of module Alphabetizer to reconstruct the circular shifts of lines.

Module Alphabetizer: The next module creates alphabetized lines of the circular shifted line invoking char and word of the circular shift module. The “genAlphabetizer()” interface is controlledby the master control. The module takes shifts and sorts the lines alphabetically using the “genAlphabetizer()” and “setchar()” interfaces.

Module Output: The “char()” and “word()” interfaces of the alphabetizing module are again invoked bythe output module to produce and print the output lines in alphabetical order and output the results to the output medium.

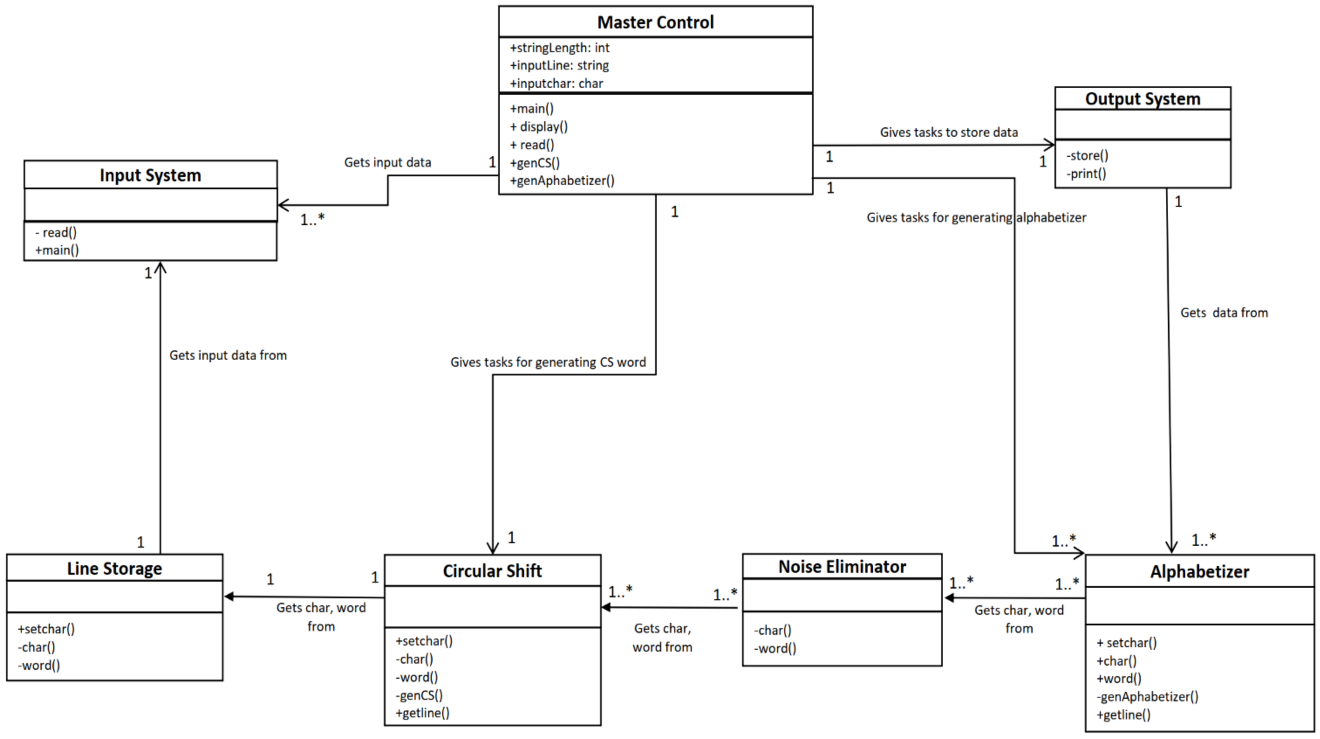
Method List of KWIC System:

Method	Description
main()	controls the master control module and invokes the read() and store() interfaces of the input and output module.
read(string line)	reads and parses a KWIC input file.
getchar(string lines,)	Adds a line at the end of the line array.
char(string lines)	Gets the line from the specified position.
word(string lines)	Adds a word at the end of the specified line.
getlines()	Gets the word from the specified position in a particular line String representing the word is returned.
genCS (ArrayList<String> lines)	Generates the circular shift of lines.
genAlphabetizer(ArrayList<String> lines)	Generates and sorts the lines alphabetically.
store()	The output is stored here and invoked by master control.
print (ArrayList<string> orderedshifts)	Prints the lines at the standard output.

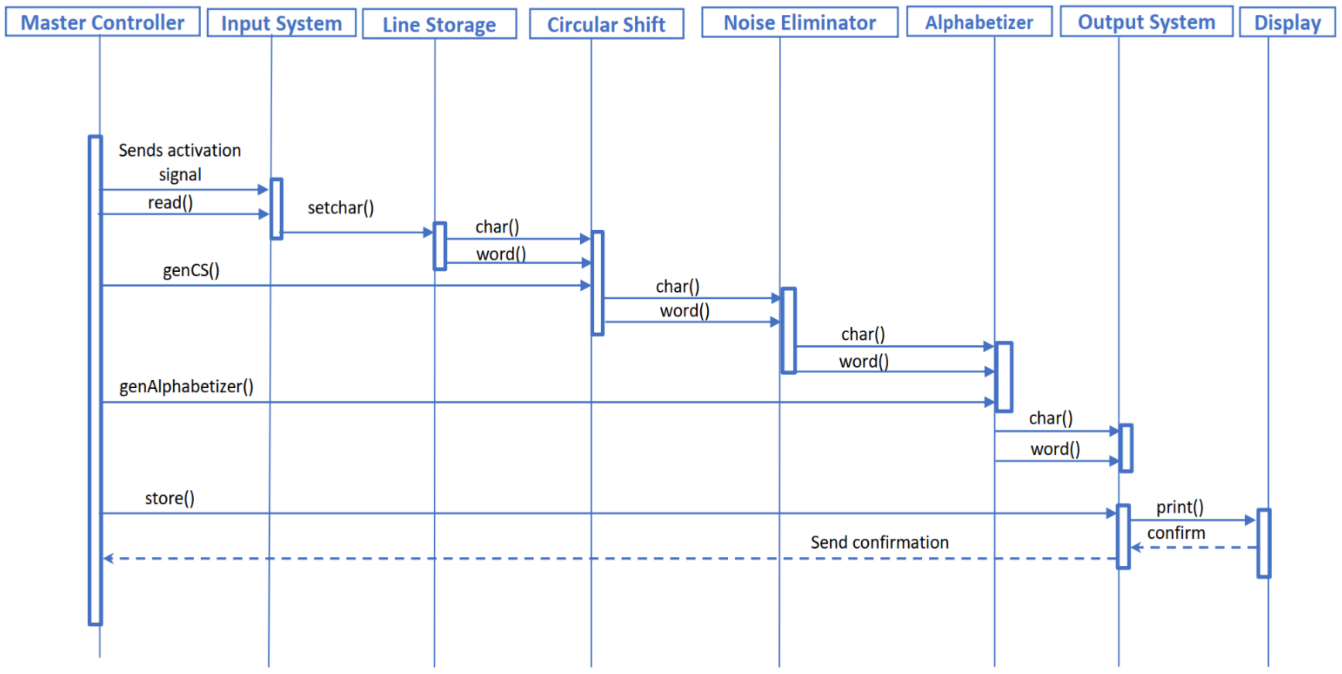
Fig. 5: KWIC System Operation List Table

Some Diagrams of KWIC System:

Class Diagram:



Sequence Diagram:



c. Cyberminer Architecture

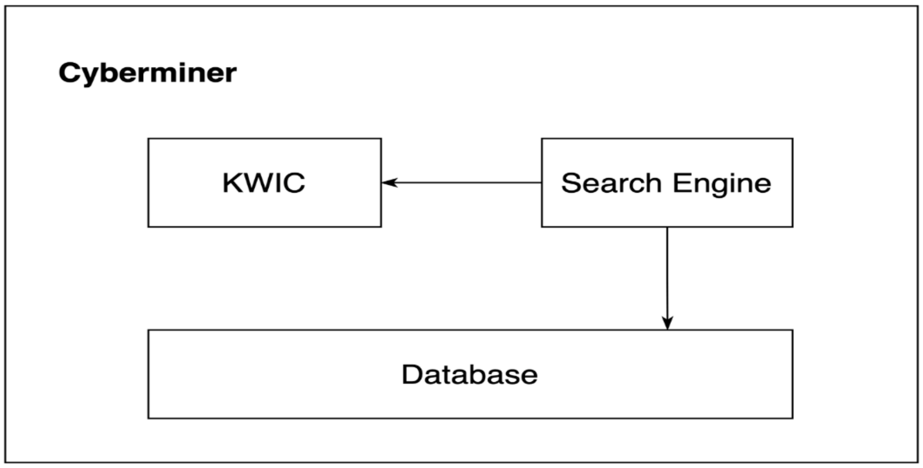


Fig. 6. High Level Cyberminer Architecture

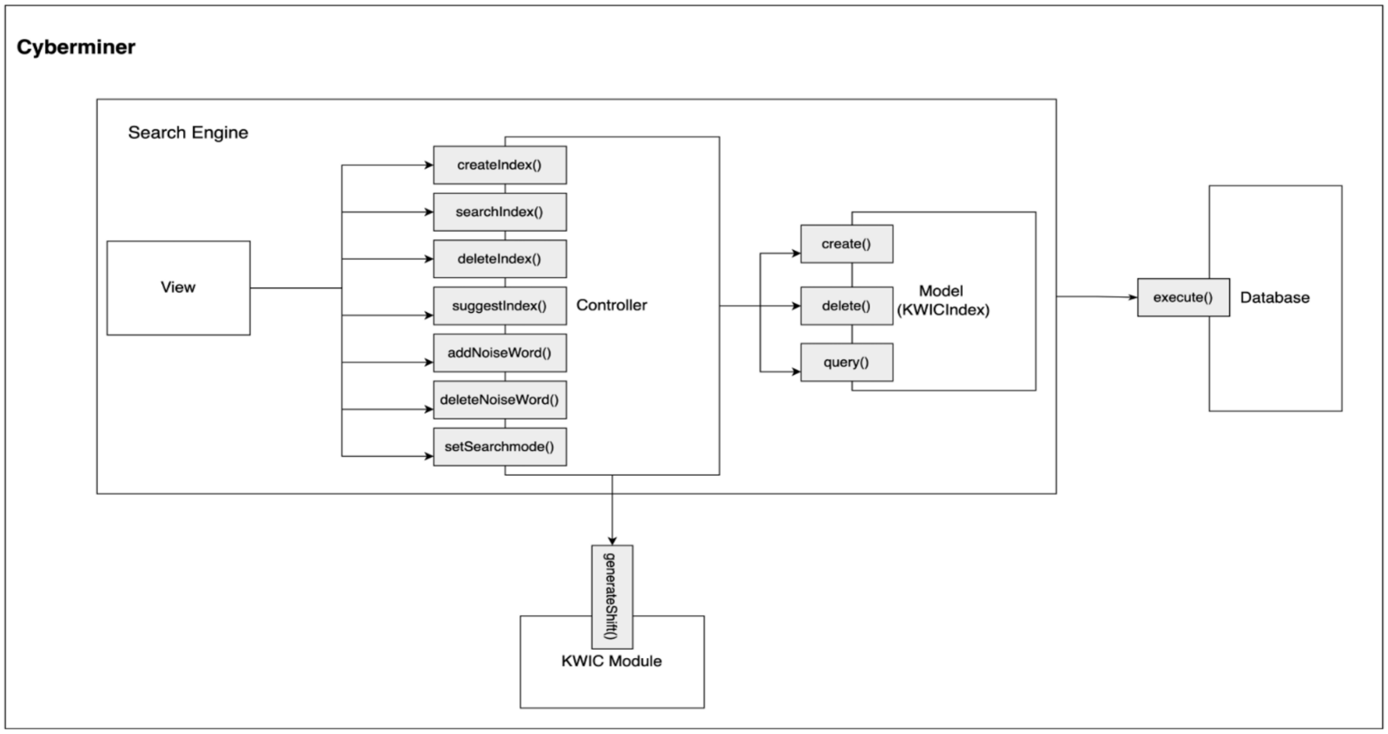


Fig. 7. Cyberminer Architecture (Detailed Design)

Components

View: Data representation is done by the view component. It generates User Interface (UI) for the user. Views are created by the data, which is collected by the model component, but these data are not taken directly but through the controller, so the view only speaks to the controller. This module will accept user input and send it to the controller. It provides a user interface for users to input searching keywords. It then takes keywords and sends them to the system backend. The system will match the keywords with the indexes stored in the index table and return the correct URL list to display in the frontend.

Controller: Controller is known as the main part because the controller is the component that enables the interconnection between the views and the model, so it acts as an intermediary. The controller collects information from view and sends them to model for modifying the searching condition or display pattern, so it just tells the model what to do. After receiving data from the model, it processes it, and then it takes all that information, and sends it to the view and explains how to represent to the user.

Model: Module is the main process component of Cyberminer. It also is known as the lowest level which means it is responsible for maintaining data. Handle data logically so it deals with data. The model is connected to the database so anything you do with data. Adding or retrieving data is done in the model component. It responds to the controller requests because the controller never talks to the database by itself. The model talks to the database back and forth and then it gives the needed data to the controller. The model never communicated with the view directly.

Repository: This component will store all data that is all URL and description the Cyberminer extract from the Internet, which is based on the indices that the KWIC module provides. Also, this module provides all data that the model component requested.

KWIC Module: This component is done at Phase 1 of project. In Phase 2, the noise word eliminator is added in original KWIC system, which increases the system sorting efficiency and reduces the indices that may disturb user searching, which enhances system usability. At last, the KWIC system is integrated with Cyberminer becoming a fully functional search engine.

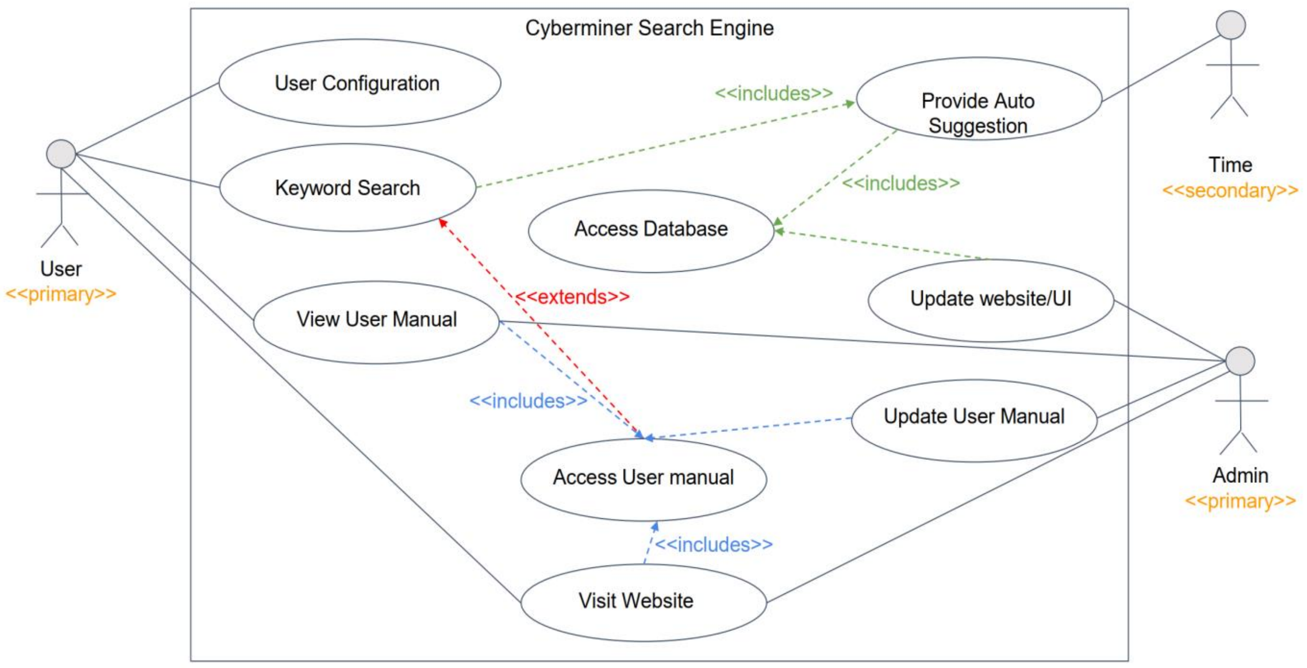
Method List of Cyberminer System:

Method	Description
createIndex(KWICIndex kwicindex)	Takes indices from interface and provides them to the KWIC system.
searchIndex(string queryString)	Takes searching key word to match with indexed stored in index table.
deleteIndex(string line)	Deletes exist indices in the KWIC system.
suggestIndex (string queryString,int limit)	Provide suggest Indices and their URL.
addNoiseWord (string noiseWord)	Gets noise word that user defined.
deleteNoiseWord (string noiseWord)	Delete noise words based on user input.
generateShift(string lines)	Sends original input and gets shifted output of KWIC system.
create (KWICIndex index)	Get results from the KWIC system and create indices for storing.
delete (string lines)	Delete input and corradiated output.
query (string query)	generate suggest output indices information.
execute (string line, string query)	Extract contained index in the repository.

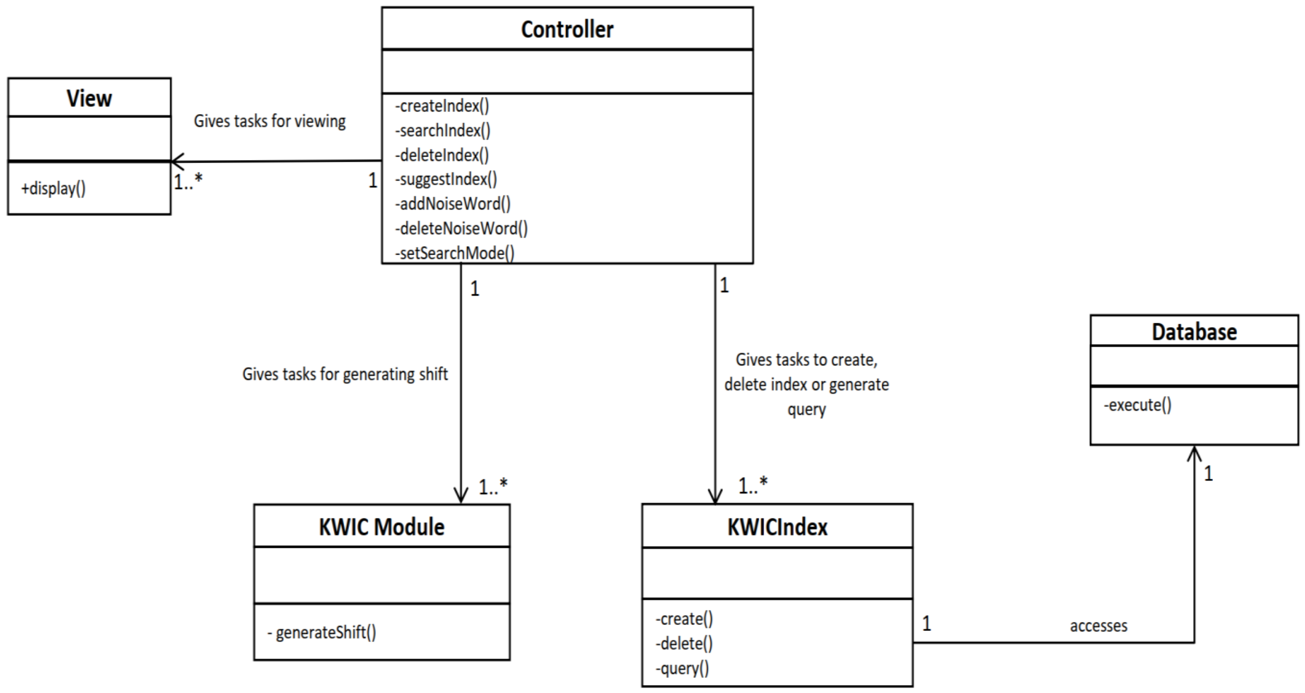
Fig. 7: Cyberminer System Operation List Table

Some Diagrams of Cyberminer Search Engine System:

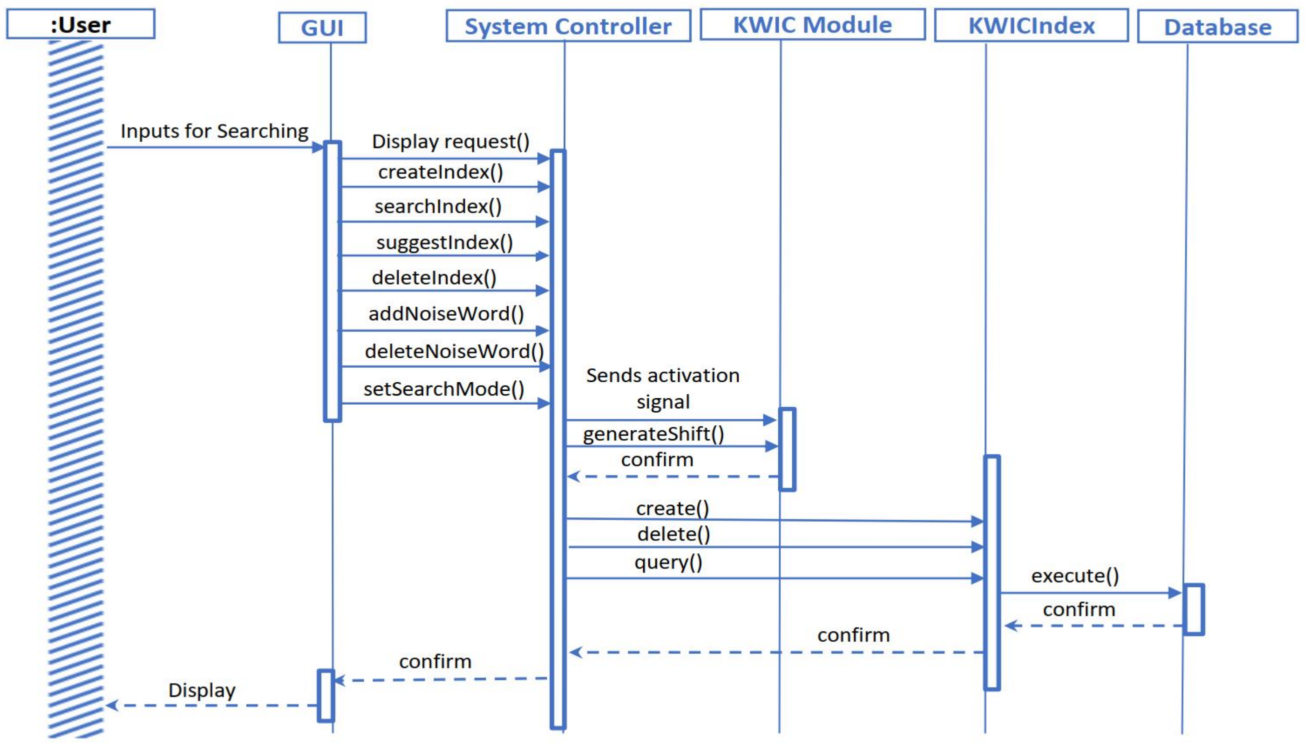
Use Case Diagram:



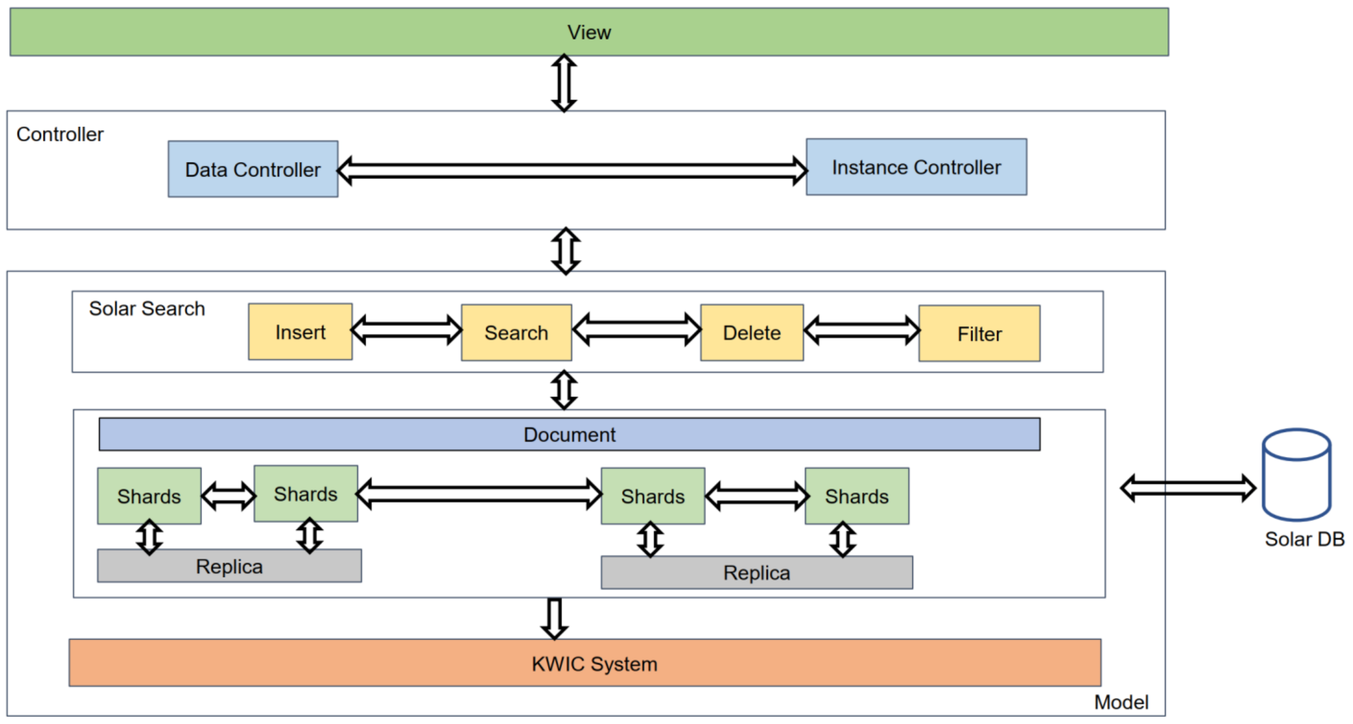
Class Diagram:



Sequence Diagram:



Design Pattern - MVC:



Interactions:

- The view and model can only communicate with the controller, there is no way that they are interacting with each other.
- The controller interacts with model and KWIC model for providing information.
- The repository can be accessed through model components.

Constraints: Each object provides an interface that permits other components to access data,only by invoking procedures in that interface.

Pattern: During running the procedure, calling other procedures in the other module is a pattern.

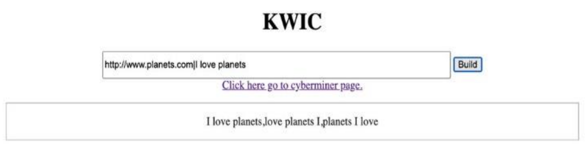
3. Specification of a website, and A Prototype

3.1 Test case

Test Case ID	Description
TC1	To check if the input can be entered from the keyboard and the result is generated from the search.
TC2	To check if the system creates output for multiple keywords search.
TC3	To check if the system navigates to a particular page when the corresponding URL is clicked from the search result.
TC4	To check if no result is given when no input is given.
TC5	To check whether there is no output when special characters are entered.
TC6	To check if the system treats the input with lowercase and uppercase letter equally.
TC7	To check if the system can navigate between pages if the search results exceed the limit on the number of results to be displayed on the single page.
TC8	To check if the system allows the user to submit multiple searches at once.
TC9	To check if the results are displayed ascending alphabetical order.
TC10	To check if the system filters out symbols that are not meaningful.
TC11	To check if the user can set number of results to be displayed per page.
TC12	To check if the system is user-friendly.
TC13	To check if the system is portable.
TC14	To check if the response time in displaying the output is minimum.

3.2 Prototype

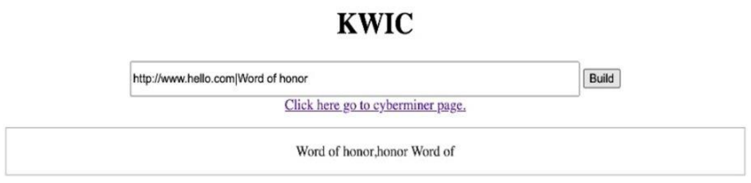
Generate Index lines:



http://www.planets.com|I love planets

I love planets,love planets I,planets I love

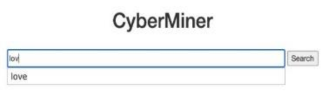
Remove noise words before being stored in DB:



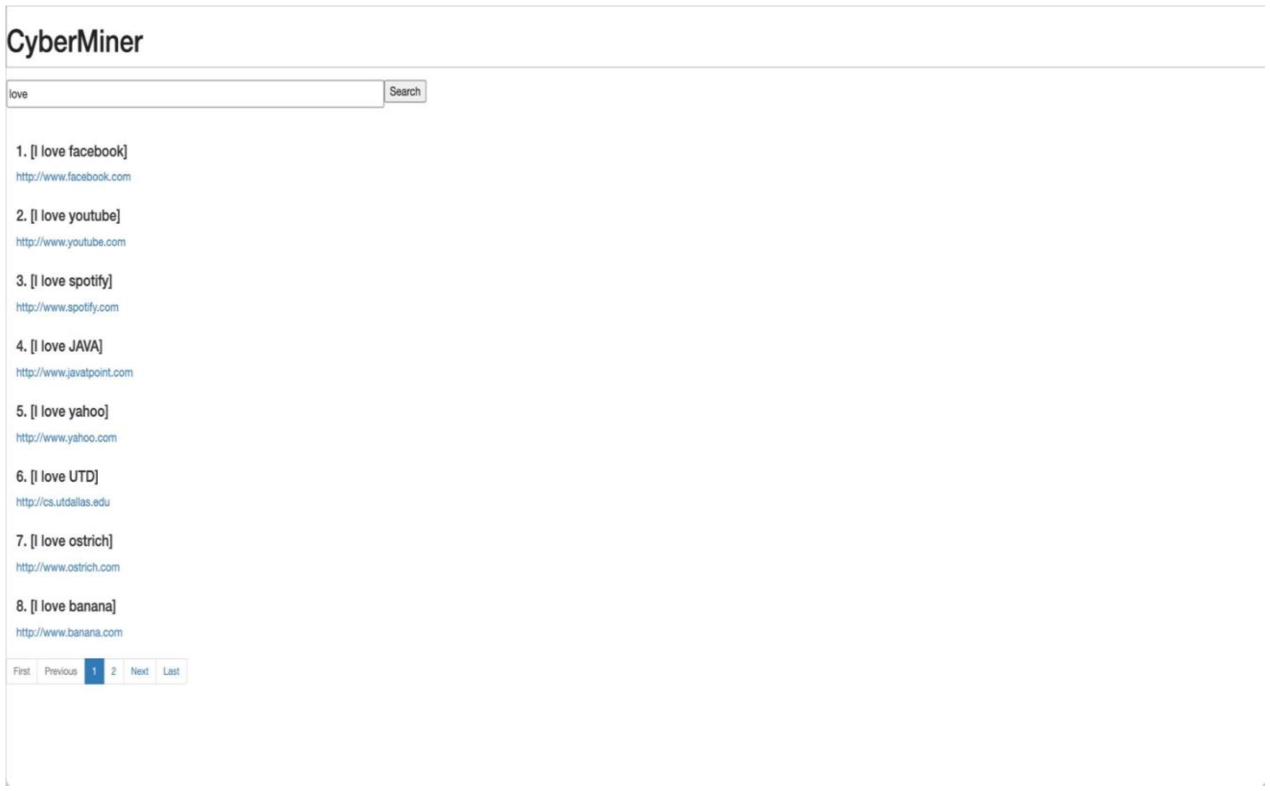
http://www.hello.com|Word of honor

Word of honor,honor Word of

Auto Complete Key words:

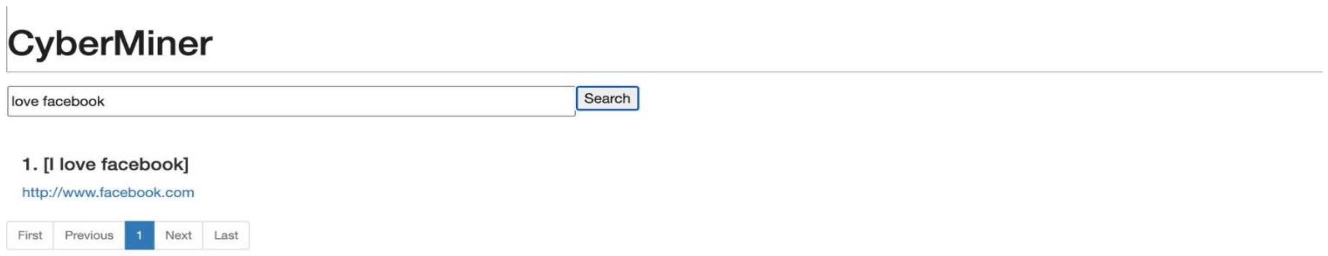


Display searching results in a paginated manner:



1. [I love facebook]
http://www.facebook.com
2. [I love youtube]
http://www.youtube.com
3. [I love spotify]
http://www.spotify.com
4. [I love JAVA]
http://www.javatpoint.com
5. [I love yahoo]
http://www.yahoo.com
6. [I love UTD]
http://cs.utdallas.edu
7. [I love ostrich]
http://www.ostrich.com
8. [I love banana]
http://www.banana.com

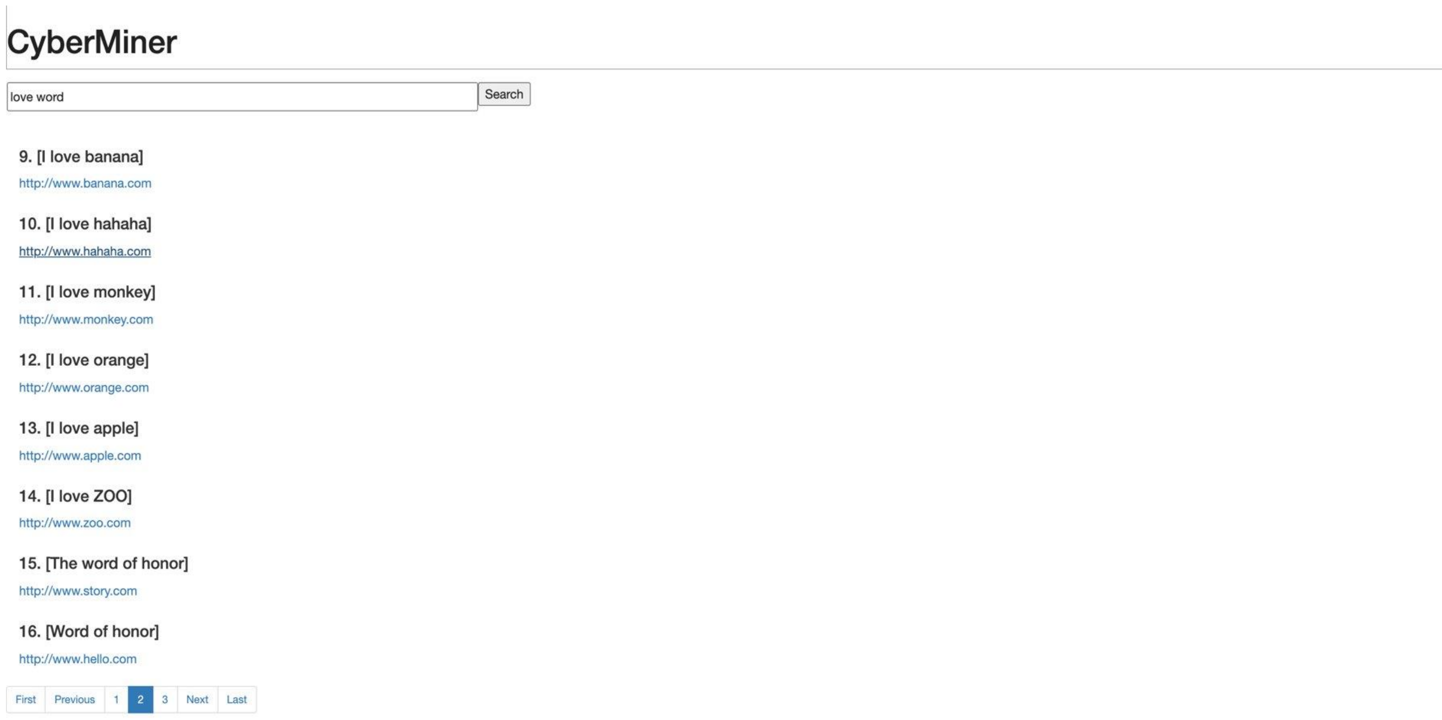
Search in AND mode:


CyberMiner

love facebook

1. [I love facebook]
http://www.facebook.com

Search in OR mode:


CyberMiner

love word

9. [I love banana]
http://www.banana.com

10. [I love hahaha]
http://www.hahaha.com

11. [I love monkey]
http://www.monkey.com

12. [I love orange]
http://www.orange.com

13. [I love apple]
http://www.apple.com

14. [I love ZOO]
http://www.zoo.com

15. [The word of honor]
http://www.story.com

16. [Word of honor]
http://www.hello.com

4. User Manual

Step 1: Enter the desired URL string and key words and click the build button to generate index lines using KWIC module.

Step 2: Click Cyberminer link to enter Cyberminer page.

Step 3: Enter key words in search box and click search button to retrieve results from database.