Assignment 1: KWIC

Code Repository URL:

Name	Mai Anh Vu	A0127046L
Matriculation Number	Chua Si Hao	A0125741L

1. **Introduction**
   The Key Word in Context (KWIC) - Assignment 1 is an index system that circular shifts the words in the lines provided. The shifted lines are alphabetically sorted and filtered to produce a list of desired titles. There are two implementations of the system created for this assignment. The *Pipes & Filters* architecture is implemented by Vu, while the *Implicit Invocation* architecture is implemented by Si Hao.

2. **Requirements**

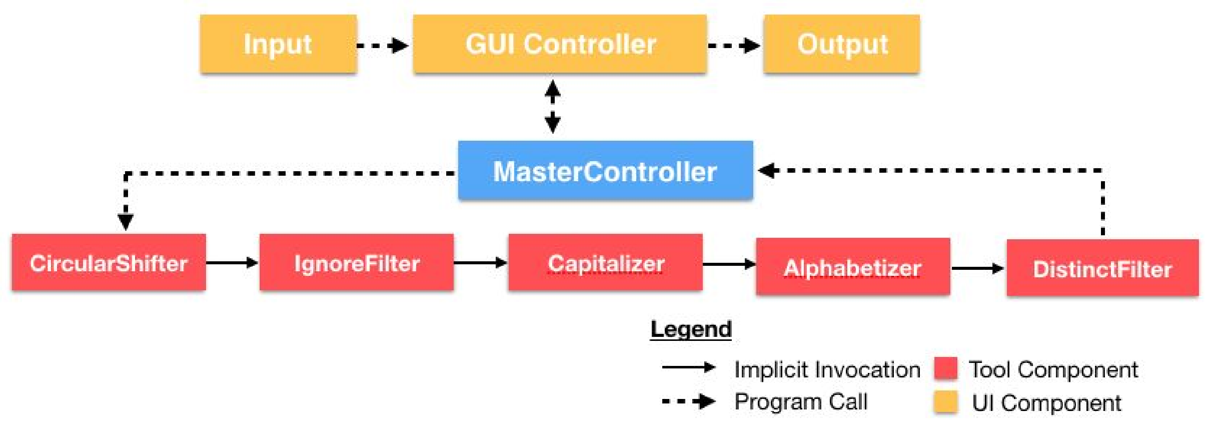
	Functional Requirement
1.	The system should be able to take in line of word(s) as input.
2.	The system should be able to take in a file with an ordered set of line(s) as input.
3.	The system should be able to circularly shift the lines exhaustively (removing the first word and appending it at the end of the line).
4.	The system should be able to order the inputs as a list alphabetically.
5.	The system should be able to have a list of words to ignore as keywords.
6.	The system should be able to remove lines that have ignore keywords.
7.	The system should be able to list only distinct titles.
8.	The system should be able to display the processed list of lines as an output.

		Non-Functional Requirement
1.	Usability	The user interface should be easy for users to use the system.
2.	Accuracy	The output must be an alphabetically sorted of circular shifted lines.
		Lines must have the first word capitalized.

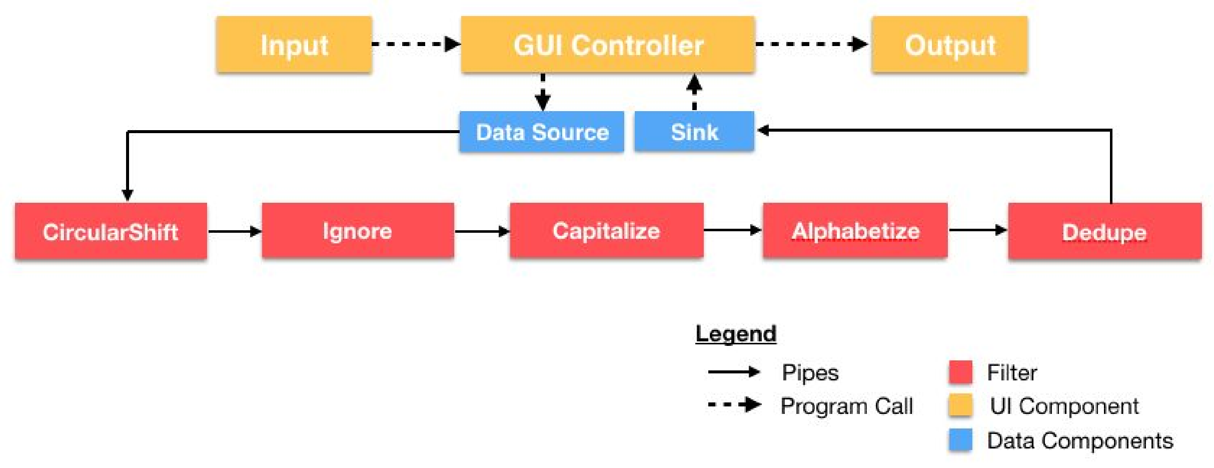
3.		Lines with ignore keyword must not appear in the output.
4.	Accuracy	The output must not have (case sensitive) duplicates.
5.	Extendable	The system must enforce separation of concern in each module.
6.		Functions should be able to be added to the system without major changes to the design of the system.
7.	Performance	The system must process each input line in < 3 seconds for a reasonable sized title.

3. Architectural Design

Pipes & Filters



Implicit Invocation



4. <u>Limitation & Benefits of Selected Designs</u>

Limitations	
Pipes & Filters	**Implicit Invocation**
● No shared data storage, extra space is needed to store (sometimes) duplicate data, and extra time is needed to reprocess data ● Limited user interactivity in the pipeline	● Unpredictability - Bindings are determined at runtime, therefore execution order is unpredictable. ● Data exchange between components are not defined and needed to be agreed upon.

Benefits	
Pipes & Filters	**Implicit Invocation**
● Filters can be added, removed, or substituted easily without affecting the rest of the application ● Execution order is easily comprehensible by looking at the pipeline ● Each filter is able to function on its own.	● High Cohesion and Loose Coupling offers increased flexibility and maintainability. ● Components work with each other but do not rely on each other to do their own job. ● High reusability - Components can be easily reused for other tasks. ● High scalability - New components can be added easily to add new functionalities.

5. <u>Any other information</u>

Implicit Invocation - Each component only allows one listener (TaskCompleteListener) to be attached, based on the assumption that it is sufficient for the implementation of this system. However, the broadcast and registering model can be expanded to allow multiple listeners and broadcast specific to the interest of the events required.

Instances of the listeners are also kept on the master controller to help reduce the effort for maintainability. Components can therefore be easily re-coupled and ordered.