**SE4433/CMSC5433 Software Architecture and Design**

**A Web Search Engine**



**1. Summary**

As software architects, your team is to architect a web search engine, Microminer, using KWIC+ which is an extension to the KWIC\* software system you have designed and implemented in Project 2. In this project, **you will use the combination client-server and MVC architecture. Particularly, you will need to implement your system with three or more-tiered C/S architecture, i.e., client, application server/web server, and database server.**

**2. Microminer - A web search engine**

**Functional requirements:** Microminer shall accept a list of keywords and return a list of URLs prefixed with their descriptions that contain all the given keywords. No noise word shall be given as a part of the list of keywords. Microminer shall use another software system, KWIC+ - a KWIC (Key Word In Context) index system, in order to efficiently store the URLs and the corresponding descriptions. KWIC+ shall accept an ordered set of lines, where each line consists of two parts: the descriptor part and the URL part:

* The URL part, whose syntax is:

URL ::= ‘http://’ {identifier ‘.’}+ [‘edu’ | ‘com’ | ‘org’ | ‘net’]

identifier ::= {letter|digit}+

letter ::= [‘a’ | ‘b’ | ‘c’ | … | ‘y’ | ‘z’ | ‘A’ | ‘B’ | … | ‘Y’ | ‘Z’]

digit ::= [‘1’ | ‘2’ | ‘3’ | … | ‘9’ | ‘0’]

* The descriptor part, whose syntax is:

{ , letter+}+

The descriptor part of any line should 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 list of all circular shifts of all lines in ascending alphabetical order <a<A<b<B<…<y<Y<z<Z, together with their corresponding URLs. No line in the output list shall start with any noise word such as “a”, “an”, “the”, “and”, “or”, “of”, “to”, “be”, “is”, “in”, “out”, “by”, “as”, “at”, “off”.

**Non-functional requirements:** The KWIC\* system shall be easily understandable, portable, scalable, and reusable with good performance. The KWIC\* system must also be user-friendly, responsive, and adaptive.

**3. The Deliverable**

Your descriptions should be elegant and comprehensible. The submission should include the following document and implementation.

• ***The process architecture*** *– management and organization of your team work: describe how your team members were divided up in carrying out your own design tasks, and why each took the particular roles. In other words, describe the essential tasks as components of your own process architecture, their essential relationships as interactions, while taking into consideration other architectural concerns. In addition, describe how your team was organized and communicated with each other, e.g., hold meeting regularly, work closely together, or mostly independent, etc*.

• **Use the 4+1 view model in this assignment.**

• **Requirement specification:** the requirement specification is incomplete. Describe any extensions or clarifications to the requirement specification. Please use the UML **USE CASE** diagram to illustrate the functional requirements.

• **Architecture specification:** You should use the combination of client server, shared data, and OO architectural style to implement the system.Please use UML ***component and deployment diagrams*** to describe both pictorially and textually the architectural style, including the components, connectors, any constrains and a discussion of advantages and disadvantages of the architecture.

• **Design specification:** Please use UML **CLASS** diagram to present your design. The design should be detailed enough so that unambiguous implementation can be achieved through the design.

• **Implementation specification:** Please use the **development view** to show the software development configuration aspects of the software system.Your program should be well documented and tested. Since shared data architecture is used, data is not allowed to be duplicated.

• **User manual:** describe how the user can access and use the system. You should describe the typical interactions between the user and the system, e.g., what are the steps the user has to follow in using the system. Use screenshots, if needed, to show how the system looks like initially as well as for subsequent steps that the user takes.