Software Requirements Specification

for

Key Word In Context

Version 2.0 Approved

Prepared by Lynn Barnett and Victoria Potvin

University of Central Oklahoma

February 4, 2015

Table of Contents

[Table of Contents 2](#_Toc412660330)

[1. Introduction 4](#_Toc412660331)

[1.1 Purpose and Scope 4](#_Toc412660332)

[1.2 Definitions and Abbreviations 4](#_Toc412660333)

[1.3 References 4](#_Toc412660334)

[1.4 Overview 4](#_Toc412660335)

[2. Overall Description 4](#_Toc412660336)

[2.1 Product Functions 4](#_Toc412660337)

[2.2 User Classes and Characteristics 5](#_Toc412660338)

[2.3 Assumptions and Dependencies 5](#_Toc412660339)

[3. Specific Requirements 5](#_Toc412660340)

[3.1 External Interface 5](#_Toc412660341)

[3.1.1 User Interface 5](#_Toc412660342)

[3.1.2 Hardware Interfaces 5](#_Toc412660343)

[3.1.3 Software Interfaces 6](#_Toc412660344)

[3.1.4 Communications Interfaces 6](#_Toc412660345)

[3.2 Functional Requirements 6](#_Toc412660346)

[3.2.1 getLines 6](#_Toc412660347)

[3.2.2 processingInput 6](#_Toc412660348)

[3.2.3 shift 7](#_Toc412660349)

[3.2.4 sort 7](#_Toc412660350)

[3.2.5 output 7](#_Toc412660351)

[3.3 Performance Requirements 8](#_Toc412660352)

[3.4 Software System Attributes 8](#_Toc412660353)

[3.4.1 Availability 8](#_Toc412660354)

[3.4.2 Security 8](#_Toc412660355)

[3.4.3 Maintainability 8](#_Toc412660356)

[3.5 Design Constraints 8](#_Toc412660357)

[3.5.1 Standards Compliance 8](#_Toc412660358)

[3.5.2 Hardware Limitations 8](#_Toc412660359)

[3.5.3 Architecture Requirements 8](#_Toc412660360)

# Introduction

## Purpose and Scope

This document outlines the requirements for a Key Word In Context system. The system takes a set of typed lines and produces a new set based on a set of algorithms. The new set will include lines which have been circularly shifted and ordered alphabetically. The purpose of this system is to prepare typed input to be used in a search engine, which will be developed at a later date.

The user interface will be simple to allow any user to utilize its functionality. The project will be web based so that it may be accessed from any computer with a standard web browser and an internet connection.

## Definitions and Abbreviations

* **KWIC\* –** From this point on we will refer to the system as KWIC\*.
* **Circular shift** – to circularly shift a line the first word will be removed and appended to the end of a line. This is done repeatedly and provides a set of lines as numerous as the number of words in a line.
* **Use Case** – a diagram and accompanying description which provides a model of how the system responds to a particular user interaction.
* **Noise Word** – a word which does not carry inherent contextual meaning and therefore cannot be used as the start of a line. At this time these words include:
  + a
  + an
  + the
  + and
  + or
  + of
  + to
  + be
  + is
  + in
  + out
  + by
  + as
  + at
  + off

## References

**UML Resources –** <http://www.uml.org/>

## Overview

This document will cover all requirements specified for this software project. Section 2 will provide an overall description of the software, including basic software functions, a description of the type of person who will use this system, and known constraints. Section 3 will outline specific requirements associated with the product, including both functional and nonfunctional requirements.

# Overall Description

## Product Functions

The product performs all functions needed to produce the circularly shifted and alphabetized set of lines from the user input. The only function available to a user is to submit input, thereby getting a set of lines. This will be referred to as the use case “getLines.”

## User Classes and Characteristics

Realistically any user without visual impairment and with a computer with a standard web browser and internet connection can use the system. It is likely to only be used by users for development of a search engine or educational purposes at the University of Central Oklahoma.

## Assumptions and Dependencies

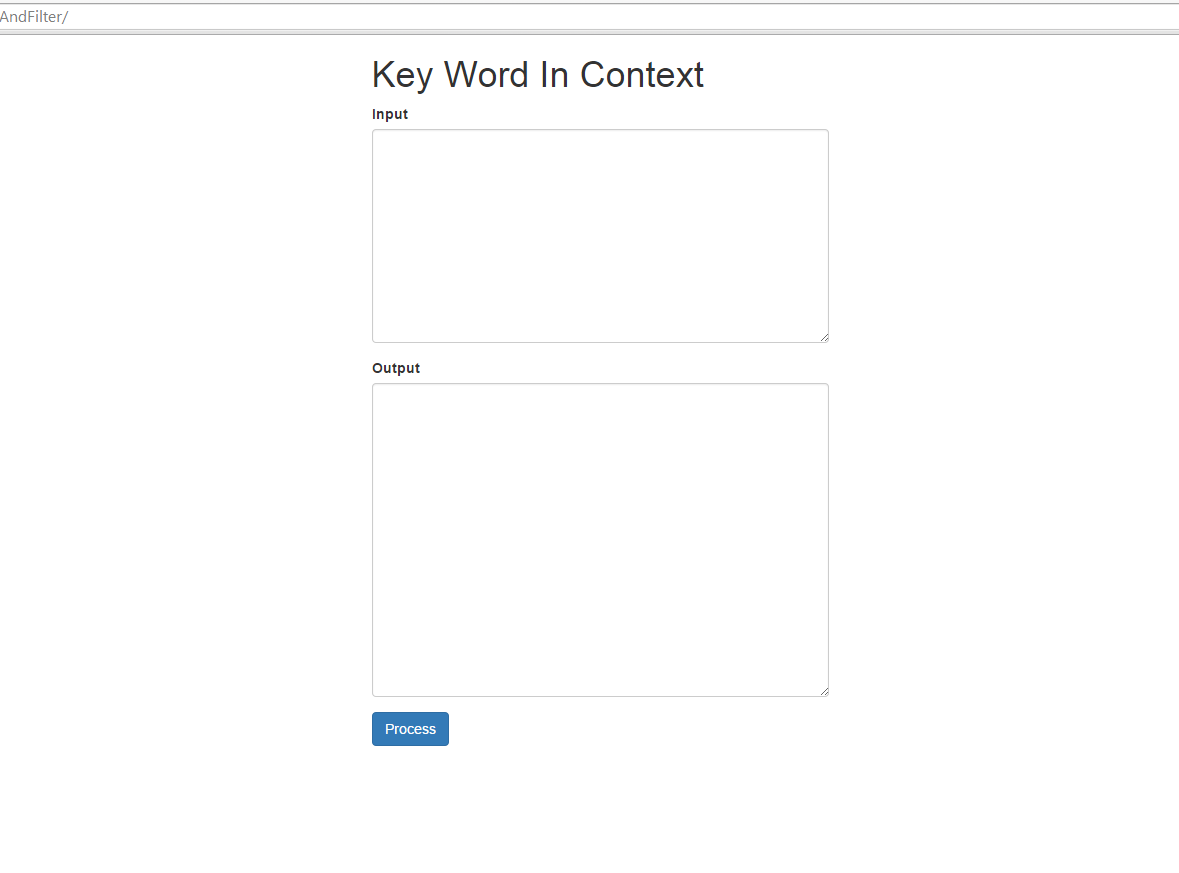
The program will be running on a web server. Since it is web based, the project is dependent on the user having an internet connection and a standard web browser.

# Specific Requirements

## External Interface

### User Interface

The user interface will be simple, composed of two text boxes and a submit button. The user will provide input in one text field and output will be displayed in the other. The user interface will be responsive, making it viewable on multiple platforms, including tablets, phones and computers.



### Hardware Interfaces

This software will run on a web server. The only requirements for the client hardware are the system requirements of the client’s chosen web browser.

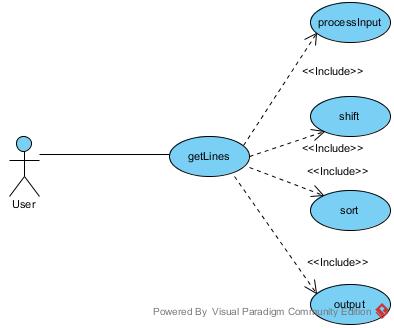
### Software Interfaces

The application will use HTML and CSS webpages to display the user interface. A webserver language will be used for calculations and processing.

### Communications Interfaces

The project will be accessed by the user though a browser utilizing HTTP. This encompasses all standard browsers, including but not limited to Internet Explorer, Safari, Firefox and Google Chrome.

## Functional Requirements



### getLines

#### Introduction

The purpose of Get Lines is to allow the user to receive an outputted set of lines created by analyzing the user’s input. This use case includes processingInput, shift, sort, and output.

#### Inputs

The user will be required to provide the set of lines they would like processed.

#### Processing

The system will read the input. It will then trigger the included use cases to continue processing.

#### Outputs

The system will output the set of circularly shifted, alphabetized lines.

### processingInput

#### Introduction

ProcessingInput is included in getLines and prepares the input to be manipulated by the application.

#### Inputs

A set of lines of text provided by the user in the getLines use case.

#### Processing

The lines of input are interpreted, divided and stored in such a way as will be easily manipulated by other parts of the application.

#### Outputs

None.

### shift

#### Introduction

Shift circularly shifts the lines.

#### Inputs

A set of lines of text provided by the user in the getLines use case which have been properly processed.

#### Processing

Each line is circularly shifted by taking the first word and putting it at the end, and then appending this line to the original list of lines. No line shall be allowed to start with a noise word.

#### Outputs

None.

### sort

#### Introduction

Sort alphabetizes the lines.

#### Inputs

A set of lines of text provided by the user in the getLines use case which have been properly processed.

#### Processing

The lines are alphabetized in ascending order, where lowercase letters have a smaller value than uppercase letters.

#### Outputs

None.

### output

#### Introduction

Output returns the lines in a form that can be viewed by the user.

#### Inputs

A set of lines of text provided by the user in the getLines use case which have been properly processed.

#### Processing

Lines are put in a form that can be viewed in a web browser.

#### Outputs

Properly formatted lines.

## Performance Requirements

The performance of the system will depend solely on the speed of the user’s internet connection and the speed of the server. No specific performance requirements have been presented to the developers.

## Software System Attributes

### Availability

The following Calculation is based off a 30-day month, this represents our estimation of Availability of the Application.

Uptime: 713 Hours

Total Time 720 Hours

Availability: 99%

This estimation is based on typical web server availability. The product is not available for use by the user if the user does not have an internet connection.

### Security

The application does not take any sensitive user data as input and therefore we do not foresee any issues with security.

### Maintainability

The client has specified that we will implement a pipe and filter architecture for the software. This architecture is known to be maintainable. Should changes be needed it is simple to delete or change a component of the system.

## Design Constraints

### Standards Compliance

We, the developers, will follow nay known best practices or implementation standards for the chosen language of implementation. We will take any necessary steps to ensure they are maintained throughout the project. The logic of our implementation will be thoroughly tested using Unit Testing.

### Hardware Limitations

We do not foresee any limitations that will affect the performance or usability of this software.

### Architecture Requirements

The client has specified that we should implement using a combination of Shared Data and OO architecture.