**Project Document**

**Table of Contents:**

1.Introduction

2.Functional Requirements

3.Non-Functional Requirements

4.Test cases

**1.Introduction**:

The introduction of the Software Requirement Engineering (SRE) provides an overview of the entire SRE with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRE. The aim of this document is to gather and analyze and give an in-depth insight of the complete **Strings and Words** by defining the problem statement in detail. The detailed requirements of **Strings and Words** are provided in this document.

The initial analysis of this project is that there is a File, which is an aggregation of many Words. Further, one can consider that there is a Counter, which keeps track of the number of different words and print them as in a specific manner. It is a matter of preference and opinion whether Counter should be function, or Counting should be implemented as an operation. For this reason, we have kept our Counter as a separate object. The basic problem statement finds only these three objects. However, further analysis for services reveals that some other mechanism is needed to check if the word is unique.

**2.Functional Requirements**:

This section contains the requirements for the project. These requirements are organized and are then refined into use case diagrams and to sequence diagram to best capture the functional requirements of the system.

* **System prompts for the file name; user enters the file name.**
* **System checks for existence of the file.**
* **System reads the words from the file.**
* **System prints the count.**

**Functionality**:

1. System should take the file name as input where the user is providing the file name as input.
2. After obtaining the file name we need to check the file existence and if that is a valid input then we must move for further steps.
3. The next functionality is to take the file input and process the information into a fine form.
4. After that checking the individual words and printing the words in a user defined manner.

**Use Case Diagram:**

Diagram

Description automatically generated

The above user case diagram will provide the information about the operations need to perform to the given input file and finalizing the required output.

**Functional Model:**

Diagram

Description automatically generated

The following diagram is the functional model of the project. Here we have clearly mentioned the operational flow of the input file i.e., Text file.

The information will be extracted from the input file and will move to the next step Get words where the strings are converted into words and these undergoes into various operations like **Tokenize, filtering,** and **finaloutput**.

These operations should appear as operations in individual or should be supported by a combination of operations. In this example, most of these processes are reflected as operations on individual files and are incorporated in the design.

**Non-Functional Requirements:**

**Usability:**

* The product shall provide a uniform look and feel between all the words.
* The product shall provide a clear information for each word in the file.

**Reliability:**

* The product is reliable to different types of input files.

**Performance:**

* The product will take initial load time depending on the input file size.
* The product highly depends on the type of input file.

**Test Cases:**

|  |  |  |
| --- | --- | --- |
| **Input Type** | **Description** | **Estimated Result** |
| Valid Input | Providing the file name  (Filename.txt). This should be in the same folder | Printing the unique words. |
| Valid Input | Providing the valid filename but it is not in the same folder. | Error. |
| Invalid Inputs | Providing the invalid filename | Error. |