Scope of work/Requirement analysis document

CITS3200 Group 11

General Goals:

* Our main goal is that our software can compare the RSID’s (revision save IDs) between two different word documents (.docx) in order to determine if any cheating or plagiarism has occurred.
* If we can successfully implement a comparison between two documents, then we will move to comparing a larger number of documents.
* A simple interface so that anyone can use the software.
* An output that is easy to understand with all relevant information, such as percentage of matching IDs and number of IDs in each document.

Current System:

* At week 4 of development, our website currently can accept two Microsoft word documents. It has no visual output, but there is some backend computation occurring to determine the percentage of matching ID values.

Proposed System:

* Overview:
  + We are making a website called DocuMatcher that can compare the hidden IDs within Microsoft word .docx files in order to assist in detecting plagiarism.
* Functional Requirements:
  + A way to upload word documents to be compared.
  + Detect common RSID codes between two (or more) documents.
  + Calculate the percentage of matching RSID codes within each document.
  + Find where in the text the matching codes are.
  + Provide the percentage and location to the user as an output.
  + Have the output as a downloadable document as well as being viewable in the browser.
  + Temporarily store, validate and then delete data after computation occurs.
* Non Functional Requirements:
  + User Interface and Human Factors:
    - Users should not need to have any sort of prior skills to access or utilise the website.
    - Any person can access the website, there will be no permissions granted to different users.
    - No training will be needed; however, we will create a user manual in order to ensure no misuse.
    - We will mitigate the chance of error through our programming; however, it is unlikely any error that occurs will cause any significant impact as the website will not store any data and will not be able to permanently manipulate the input data.
    - All user actions should be taking place within the web browser.
  + Documentation:
    - We will create a user manual to ensure proper use.
    - Comments within our code to help anyone understand how it works, this will only be relevant to other people with programming experience.
    - This scope of work is made to state our initial understanding of the project and to put into writing all the necessary actions we will need to take. This will be useful for the members of the group working on the assignment as well as the group auditor.
    - We have made a skills and resource audit to make sure we all understand what knowledge is relevant to create the software.
    - A risk register will help the group members to avoid potential risks
    - Project acceptance tests will be used to finalise the project, they help to ensure that the project has been successful
    - User stories will be created to make sure that the most important features have been implemented when we look back during later parts of the project.
  + Hardware Consideration:
    - Our website should be usable on any device with internet access.
    - The characteristics of the user’s device should be effectively irrelevant. Computation should be done on a backend server, if not it still shouldn’t be particularly taxing, so any reasonably functioning device should be sufficient.
  + Performance Characteristics:
    - Ideally the website would process a comparison of two documents within a few seconds. Until a maximum document allowance is decided it is too difficult to put an overall cap on speed.
    - As previously decided, the size constraints are yet to be decided and will be figured out as the project develops.
  + Error Handling and Extreme Conditions:
    - The system will only allow .docx files to be uploaded in order to avoid errors, if less than two documents are uploaded then an error message will be displayed to the user explaining the issue.
    - Errors that will be encountered later in the development will be dealt with effectively at the point of occurrence.
    - All errors should provide the user with a notice of why the error occurred.
    - There should not be any extreme situations in our website.
  + System Interfacing:
    - The input will be coming from the user’s device
    - The output will be displayed in the browser, but an option for the user to download the output will be available.
    - The input must be a .docx file, which is made known to the user, the output format will be decided later (perhaps .txt or .pdf).
  + Quality Issues:
    - The website should be simple to use, which means there should rarely be any errors.
    - The website should be available at all times of the day.
    - Portability is not particularly important as most use will be on a computer’s browser, but still should be usable on any device.
  + System Modifications:
    - The only part of the system that may need changing in the future is the number of documents that are allowed to be uploaded at once, currently it is capped at two.
    - The comparison feature is yet to be programmed, so that will need to be created.
  + Physical Environment:
    - The physical environment is irrelevant to the system, assuming the user has an internet connection.
  + Security Issues:
    - Users will not need to make accounts, so no personal data will be saved.
    - The files that are uploaded will be stored only temporarily and will be deleted once the session is closed.
  + Resource Issues:
    - We will manage the maintenance of the system if any is required.
    - Our client will host the website himself.
* Constraints:
  + There should not be many constraints on the development of the system. The programming languages used are flexible.
* Use Case Model:
  + A diagram of a document

    Description automatically generatedThe use case model for the website is very simple as there are a limited number of features.
* Interface:
  + This is what the main page looks like currently, there should not be many changes made by the time of launch. We do not currently have an output interface.

