Report on Text Similarity Checker (Plagiarism Detector)

Introduction

In the digital era, plagiarism detection and text similarity analysis play a vital role in academia, research, and content creation. With the growing availability of online resources, ensuring originality has become increasingly important. This project, *Text Similarity Checker*, is designed to compare two documents, calculate their similarity score, and highlight overlapping content. The system provides an automated and efficient way to identify potential plagiarism, thus supporting integrity and originality.

Abstract

The *Text Similarity Checker (Plagiarism Detector)* leverages modern information retrieval and text analysis techniques to determine the similarity between two documents. Using **Apache Lucene** for indexing and searching, combined with **iText7** for PDF generation, the system processes input files, removes noise such as stopwords and punctuation, and applies cosine similarity for accurate comparison. The solution also highlights matching content and generates a detailed report with a similarity percentage, exportable in PDF or text format. This ensures both precision and usability in real-world applications.

Tools Used

1. **J2EE (Java 2 Platform, Enterprise Edition):** Used to build and manage the server-side components of the application, ensuring scalability and robustness.
2. **Apache Lucene:** A powerful text-search engine library used for tokenization, indexing, and efficient similarity computation.
3. **iText7:** A PDF library utilized for exporting the similarity analysis results in a structured and professional format.

Steps Involved in Building the Project

1. **Reading Input Files:** The system accepts two documents as input for comparison.
2. **Tokenization and Preprocessing:** Both documents are tokenized into words. Stopwords and punctuation are removed to eliminate noise.
3. **Indexing and Vectorization:** The processed text is indexed using Apache Lucene, and term vectors are generated.
4. **Cosine Similarity Computation:** The cosine similarity algorithm is applied to calculate the degree of similarity between the two text vectors.
5. **Highlighting Matches:** Matching phrases are identified and marked to provide a clear visualization of overlaps.
6. **Displaying Results:** The final similarity percentage is presented to the user in an intuitive format.
7. **Exporting Reports:** The results, including highlighted matches and similarity scores, are exported in PDF or plain text using iText7 for professional documentation.

Conclusion

The *Text Similarity Checker* successfully demonstrates an efficient approach to plagiarism detection and document comparison. By integrating **J2EE, Apache Lucene, and iText7**, the system provides reliable similarity scoring, clear highlighting of matched content, and professional reporting capabilities. This project can be extended to support larger datasets, multiple file formats, and real-time analysis, making it a valuable tool in educational, research, and professional domains.