Text Similarity Checker

# 1. Project Overview

The Text Similarity Checker is a Java-based application designed to compute the similarity between two text documents. It uses a hybrid approach combining cosine similarity, Jaro-Winkler similarity, Levenshtein distance, and synonym expansion to provide a more accurate and context-aware similarity score.

# 2. Features

- GUI with a clean and intuitive interface.

- Input support for .txt files (default) and manual text input.

- Synonym-aware cosine similarity using a large custom synonym file or the Datamuse API.

- Hybrid similarity score combining fuzzy matching techniques.

- Highlighting of matching phrases and words.

- Export results as .txt or .pdf files (user-defined save path for PDF).

- Support for customizable synonym list via synonyms.txt.

# 3. Folder Structure

TextSimilarityChecker/  
│  
├── input/ # Contains input text files and synonym list  
│ ├── doc1.txt  
│ ├── doc2.txt  
│ └── synonyms.txt  
├── src/ # Java source code  
│ ├── Main.java  
│ ├── MainGUI.java  
│ ├── TextProcessor.java  
│ └── ExportUtils.java  
│  
├── README.md # Project readme  
└── .gitignore # Git ignored files list

# 4. Technology Stack

- Language: Java

- GUI: Swing

- Similarity Libraries: Apache Commons Text, Apache Lucene

- PDF Export: iText

# 5. How it Works

1. Tokenization: The text is tokenized using Lucene's StandardAnalyzer.  
2. Synonym Expansion: Each token is expanded using a synonym map built from synonyms.txt or the Datamuse API.  
3. Frequency Mapping: A term-frequency map is created for each expanded token list.  
4. Cosine Similarity: Calculates similarity based on frequency vector dot product.  
5. Hybrid Matching: Applies Jaro-Winkler and Levenshtein distance with a configurable threshold.  
6. Final Score: Cosine and hybrid scores are computed and displayed alongside highlighted matching terms.

# 6. How to Run

1. Open the project in any Java IDE (like IntelliJ IDEA or VS Code with Java support).  
2. Ensure dependencies (commons-text, lucene-core, lucene-analyzers-common) are included in the classpath.  
3. Modify or add your own .txt files inside the input/ directory.  
4. Run Main.java to launch the application.

# 7. Exporting Results

- PDF Export: File chooser is shown for the user to select destination.

# 8. Example Input Sentences

Text 1:  
A fast brown fox leaps over a lazy dog.  
This phrase is commonly used to check fonts and typing.

Text 2:  
The quick brown fox jumps over the lazy dog.  
This sentence contains every letter of the alphabet.

Cosine Similarity: 68%  
Hybrid Similarity: 81.19%  
Matching Terms: [fox, brown, dog, typing, alphabet, ...]

# 9. Future Improvements

- POS tagging and phrase-level similarity.

- Sentence embeddings using pre-trained NLP models.

- More advanced synonym detection using WordNet.

- Web-based interface with backend processing.