**Simplifying Text with Word Embeddings**

@author Declan O’Meara G00439376@atu.ie

@version Java 21

# Project Description

A console based Java text simplification application that converts complex text into simpler alternatives using the Google-1000 most common words list and GloVe word embeddings. The application employs virtual threading for efficient processing and offers multiple similarity calculation strategies and multiple output options.

# To Run

From the console run: java -cp ./simplifier.jar ie.atu.sw.Runner

Navigate through the menu options to:

1. Set word embeddings file path
2. Set Google-1000 words file path
3. Set input text file path
4. Configure output file path (default: ./out.txt)
5. Select similarity calculation strategy
6. Choose output format

# Key Features

## Core Architecture:

* Advanced Concurrent Processing:
  + Virtual threaded processing of word embeddings
  + Thread-safe data structures (ConcurrentHashMap)
  + Atomic operations for counters
* Design Pattern Implementation:
  + Strategy Pattern for similarity calculations
  + Strategy Pattern for output formatting
  + Extendable design for future similarity methods and output strategies due to interface driven design and modular component structure
  + High cohesion and loose coupling

## Key Features:

* Multiple Vector Similarity Algorithms:
  + Cosine Similarity (default)
  + Dot Product Similarity
  + Euclidean Distance
  + Manhattan Distance
* Choice Output Options:
  + Standard file and console output with formatted headers
  + JSON structured output
  + ANSI color-coded interface
  + Dynamic border generation
* Robustness:
  + Comprehensive error handling to manage invalid inputs or misconfigurations with feedback to client
  + Graceful fallback to default configurations when required data is missing or incorrect.
* Processing and Management:
  + Comprehensive file validation
  + Detailed processing statistics:
    - Words simplified count
    - Words in Google-1000 count
    - Words not in embeddings count
  + Real-time progress feedback
  + Interactive help system
* Performance Optimizations:
  + Parallel processing with virtual threads
  + Optimized vector calculations
  + Memory-efficient data structures