 # CSI4107_Assignment1 **# Information Retrieval System README**

Team Members
- Member 1: Sattar Abdul, Student Number: [300156257]
- Member 2: Stephen Rioux, Student Number: [300175532]
- Member 3: Osa Ikhinmwin, Student Number: [300211931]

Task Division (33% each member)
The work was divided equally between the members.
- Member 1: Responsible for the 3 steps (processing, indexing, retrieval and ranking)
- Member 2: Responsible for outputs and optimization
- Member 3: Responsible for setting up environments and overall project

Functionality Overview
Our Information Retrieval (IR) system performs document indexing and query processing. The IR system streamlines the process from document preparation to retrieval. It begins by cleansing texts through tokenization, eliminating common words, and standardizing terms. An inverted index is then constructed to facilitate quick document access. The system calculates TF-IDF scores to evaluate term significance across the document corpus. For queries, it retrieves relevant documents by comparing their TF-IDF vectors with those of queries using cosine similarity, thus ranking documents by relevance. This comprehensive approach ensures efficient and accurate document retrieval and ranking, enhancing the IR system's effectiveness.

Running the Program
1. Install Python and necessary libraries (`nltk`).
2. Place documents in the `AP_collection/coll/` directory or adjust the path
3. Put the stop words and topics docs in the same directory
4. Run `main.py` with the command `python main.py`.

Algorithms and Data Structures

Preprocessing (Step 1)
- ****Tokenization****: Splitting text into individual terms using regular expressions.
- ****Stop Word Removal****: Filtering common words using a predefined list loaded from `StopWords.txt`.
- ****Stemming****: Reducing words to their base or root form using NLTK's PorterStemmer.

Indexing (Step 2)
- ****Inverted Index****: Implemented using a defaultdict of dictionaries in Python, mapping terms to their document IDs and frequencies.

- **Term Frequency (TF)**: Counting the number of times each term appears in each document.
- **Document Frequency (DF)**: Calculated during inverted index creation by tracking the number of documents each term appears in.
- **TF-IDF Computation**: Employing the TF and DF values to calculate the importance of each term in the document corpus, which is used for ranking.
- **Document Vector Lengths**: Calculating the Euclidean norm of each document vector to normalize the cosine similarity scores during retrieval.

Retrieval (Step 3)
- **Query Vectorization**: Transforming the query into a vector form based on the IDF scores from the indexed corpus.
- **Cosine Similarity**: Incrementally computed as query terms are processed to measure the relevance of documents to the query.
- **Ranking**: Sorting retrieved documents by their cosine similarity scores using a heap or sorted list data structure for efficiency.

Evaluation Measures
- **Precision and Recall**: Standard evaluation metrics used to assess the effectiveness of the IR system. We used the trec_eval. A file is included with the results from that evaluation.

Vocabulary and Results
- Vocabulary Size: 159860
- Sample Tokens: ['doc', 'docno', 'ap880212', '0001', 'fileid', 'ap', 'nr', '02', '12', '88', '2344est', '1st_line', 'vietnam', 'amnesti', '0398', '2nd_line', '0411', 'head', 'report', 'saigon', 'offici', 'releas', 'educ', 'camp', 'datelin', 'bangkok', 'thailand', 'text', '150', 'offic', 'ofic', 'overthrown', 'south', 'vietnames', 'govern', '13', 'year', 'detent', 'news', 'agenc', 'saturday', 'hanoi', 'monitor', 'specif', 'figur', 'freed', 'friday', 'ex', 'cabinet', 'minist', 'deputi', '10', 'gener', '115', 'field', 'grade', '25', 'chaplain', 'quot', 'col', 'luu', 'ham', 'director', 'nam', 'ha', 'say', '700', 'held', '1', '014', 'announc', 'communist', 'mark', 'tet', 'lunar', 'feb', '17', 'foreign', 'journalist', 'deleg', 'australia', 'friendship', 'associ', 'attend', 'ceremoni', 'lt', 'gen', 'nguyen', 'vinh', 'nghi', 'command', 'armi', 'corp', 'tran', 'duc', 'minh', 'infantri', 'school', 'express', 'gratitud', 'human']

Performance Evaluation
- Mean Average Precision (MAP) for 50 test queries:
map   all   0.0000

- We tried different techniques but couldn't improve the map very much