Query Processor

System Design Overview:

The Query Processor system is designed to process documents using Information Retrieval techniques based on TREC (Text REtrieval Conference) format. The system involves several components:

Text Tokenization and Preprocessing:

- The TextTokenizer class handles text tokenization and preprocessing tasks such as stemming and removing stopwords.
- It utilizes regular expressions and NLTK's Porter Stemmer for processing text.

Document Reading and Parsing:

- The DocumentReader class reads and parses documents in the TREC format.
- It extracts document IDs, text content, tokenizes the text, and preprocesses it for further indexing.

Query Extraction:

- The QueryExtractor class extracts queries from TREC topics.
- It identifies query numbers and titles from the provided text and creates a mapping of query numbers to their titles.

Indexing and Retrieval:

- The IndexHandler class manages indexing tasks.
- It creates both forward and inverted indices for documents and implements search functionalities.

Term Weighting and Normalization:

Term Weighting:

- The system uses the TF-IDF (Term Frequency-Inverse Document Frequency) scheme for term weighting.
- During the indexing process, it calculates TF-IDF scores for each term in documents to represent their importance.

Normalization:

- For both documents and queries, normalization involves logarithmic scaling of term frequencies.
- Normalization of term frequencies is done to prevent bias towards longer documents and frequent terms.

Implementing Cosine Similarity

- **Vector Representation:** Each document and query is represented as a vector in a high-dimensional space, where each dimension corresponds to a term's TF-IDF weight.
- **TF-IDF Calculation:** TF-IDF scores for terms in documents and queries are calculated during indexing and query processing, respectively.
- **Vector Normalization:** Both document and query vectors are normalized to unit length, ensuring comparable magnitudes for cosine similarity.
- **Cosine Similarity Calculation:**
- Document-Query Similarity: For a given query, cosine similarity is computed between the query vector and each document vector.
- **Scoring and Ranking**: Documents are scored based on their cosine similarity values with the query, enabling ranking of documents according to relevance.
- **Ranked Retrieval:** Retrieval of documents involves sorting them in descending order based on their cosine similarity scores with the query.

Data Structures and Classes:

TextTokenizer Class:

- Responsible for text tokenization and preprocessing.
- Utilizes regular expressions and NLTK's Porter Stemmer.

DocumentReader Class:

- Handles document reading and parsing tasks.
- Extracts document IDs, text content, tokenizes, and preprocesses text.

QueryExtractor Class:

- Extracts queries from TREC topics.
- Identifies query numbers and titles for further processing.

IndexHandler Class:

- Manages indexing and retrieval tasks.
- Creates forward and inverted indices for documents and performs search operations.

Conclusion:

The Query Processor project is a straightforward system designed to handle document retrieval tasks following the TREC format. It performs text processing, indexing, and search operations using TF-IDF weighting for relevance ranking.