The application consists of a two Python files, one that implements Porter Stemmer Algorithm (obtained online) and the other that performs both indexing and querying. Below is the design description of the code that performs indexing and querying:

First of all, all the documents (.txt) files are read from the dataset whose location is passed as an argument to an in-built method in the “os” module of Python. Then tokens are generated for all files and normalized through a series of steps involving removal of punctuations, conversion to lower case, exclusion of stopwords and stemming to obtain the terms that will be saved in the dictionary. Posting list is created for each term and term frequencies and idf value is stored. Then the user is prompted to enter a query and top 20 results are returned. Ranking is based on the Vector Space model of Information Retrieval.

Data Structures used:

Dictionary: This data structure is used to store the terms in the dictionary and their posting lists for fast retrieval of data corresponding to a given term. It is also used to map doc id to the title of the document and to store cosine scores of each term in the query with respect to a document.

Lists: This data structure is used to store the posting lists corresponding to a given term. Lists have dynamic length so they are efficient in terms of memory.

Tuple: this is used to store the document id and its normalized term frequency in the posting list. Since the corpus does not change with time, so the term frequencies remain constant and are better stored in an immutable data structure.

Main variables used in the code:

1. dictionary={} – Dictionary that maps each term to a list of tuples, used to store the terms and their posting lists.
2. corpus={} - store doc ids and titles.
3. idf = {} - used to store the idf for each term.
4. q\_dict ={} - Dictionary that stores the normalized tf-idf score for each term in the query.
5. score\_dict = {} – stores the cosine score for all relevant docs with respect to each term in the query.

Python modules used:

1. nltk – generate tokens from text and get English language stop words.
2. os,io,glob – read text files from the dataset.
3. math- perform log and square root operations.
4. operator- sort the list of documents in decreasing order of ranking for a search query.
5. Porter Stemmer’s Algorithm Python implementation which is available as open source at: http://tartarus.org/~martin/PorterStemmer/python.txt