Information Retrieval Assignment 1
Design Document

Introduction:
A text-based information retrieval system using tf-idf and vector space model.

Corpus:

Description:
All the course handouts of the current semester from CMS.
Total number of documents: 183
Each logical document is a CMS handout file.

All handouts were manually downloaded and converted to plain text documents with ascii-encoding and to the pdf format to distribute the documents easily.

Phase 1:

Technologies & Languages:

The entirety of phase 1 has been implemented using python 2.7 for any processing and MySQL 5.7 to store the relevant data.

Tokenization and Normalization:

The word tokenizer and the porter stemmer in the nltk library for python has been used to tokenize and stem the raw text.

Regular expressions were then used to split any tokens that weren't split by the word tokenizer. This includes splitting by hyphens among other things.

The stopwords list started out with the nltk stopwords. Words that occur in more than 60% of the documents were carefully added making sure no unintended words slip out. Many more tokens were hand-picked and added.

Dictionary Construction:

A sparse matrix that stores the term frequency for each term and document combination was created in MySQL.
Postings list for each token was then implemented as a string of document numbers of documents containing the token separated by commas. These postings lists were appended as a column to the aforementioned matrix after the document frequencies and idf values.

Index Construction:

Vector space model was used to build the index.

Document vectors have been calculated using the tf-idf model and stored in another MySQL relation.

A .csv file containing CMS course ID, name, full name and URL has also been stored in the database.

Phase 2 (Querying and Retrieval):

Through the console:

Querying and retrieval is done using a python script that takes user input from the console. This input is then tokenized using regular expressions and stemmed using the same porter stemmer used in index construction.
Regular expressions were also used, as a special scenario with this dataset to identify course names (like CS F469) when they aren't separated by whitespace(s) (like csf469).

The posting lists for these tokens are then retrieved from the database.
From these posting lists, document numbers of documents containing all the query tokens have been collected into a list and those of documents with at least one of the query tokens into another list.

If at least one document contains all query tokens, then only these documents are ultimately displayed. If not, the documents from the other list are displayed.

The relevance order is obtained by getting the score of each document using vector space model from the database.

These document numbers are then used to retrieve the respective course codes, names and CMS URLs also from the database and then printed to the console.

Through a webpage:

A php script was made that uses the same method as the aforementioned python script and hosted at the localhost (172.16.117.42 as of 23:16 27th Sep) on an apache server so that the handouts can directly be served. We hope this is out of box enough.