Semester Project Report

Design

The project is divided in three main part: the crawling done with `Nutch`, the indexing and retrieval of document with `Lucene` and the rest API serving the data implemented with `SpringBoot`. The main application is divided in three main classes:

1. Indexer
2. Retriever
3. QueryController

The `Indexer` has the task of parsing the dumped file generated by `Nutch`. The information I save in the document storage are the title, the content and the url. The parsing is pretty simple since I don't have to exclude almost any document at this point because the task to perform a *Focused Crawling* is handled directly by `Nutch` as I will explain in my implementation choices.

The `Retriever` is called by the `Query Controller` and take the issued query and generate a list of query results.

The `Query Controller` receive the request for a issued query and return an object containing all the information to display the results list.

Implementation Choices

The biggest effort for the project was actually in adding the *Focused Crawling* feature. In the end I manage to force `Nutch` to crawl exactly the articles I wanted by starting from a well defined seed and then forcing `Nutch` to follow only articles links by restricting the crawl with some regex to the configuration file `regex-utlfilter.txt`.

The starting seed is a collection of links that conduct to the list of articles published by 10 different authors: Adam Payne, Barbara Tasch, Lianna Brinded, Ben Moshinsky, David Scutt, Thomas Colson, Will Martin, Lindsay Dodgson, Rachel Gillett, Hannah Roberts.

The biggest problem is that I was able to crawl only from one blog: `http://uk.businessinsider.com/`, because it was the only blog I found that had a simple structure, a well defined page with the articles of one author, simple links that permitted a selection by regular expressions and that was not blocking me during the crawling.

I choose to implement a `Spring Boot` application to handle the requests and serve the results because of simplicity. Before the server is started the indexing is performed by `Lucene` and all the code related to the indexing and retrieving of documents is based on the tutorial had in class plus the usage of some snippets from online tutorials.

The *UI* is based on a free template[1] and is kept clean by being as minimal as possible but with all the information needed to decide if a document is relevant or not.

[1] Miminium Bootstrap template by akivaron: `https://github.com/akivaron/miminium`

Evaluation

By asking three student to evaluate my search engine the main points that came out were:

Positive Points:

- Good feeling with the *UI* described as simple, easy to understand and with a clean results view
- Easy to understand the pertinence of a document by the content snippet.

Negative Points:

- Since the results are blog post users expected to see the date of the post.
- In many different queries the first result was not the most relevant one for the user
- The name of the author of a post is not visible in the results view.

Submitted files

In the `Nutch` folder can be found all the modified configurations files, the initial seed, the crawl folder and the final dump document.

In the `IR-SpringBoot-Lucene` folder there is the source code for the main application that can be directly opened as a `IntelliJ IDEA` project. The webapp run on `localhost:8090`.