**OOA**

1. **Finding Classes & Objects**

**Nouns**

* Documents
* Search Engine
* Single-Word Query
* Multi Word Query
* Setup
* Inverted Index
* CLI
* Collection of Docs
* Stoplist

**Roles Played**

* Searcher
* Single Word Query Builder
* Stop list Checker
* Multi word Query processor
* Command Line Processor

1. **Finding Attributes**

**Search Engine**

* N/A, retrieves attributes from other classes

**CLI**

* N/A, retrieves attributes from other classes

**Setup**

* A collection different groups of pointers to a certain document based the word in the query (responsible for setting up said collection)
* A List of Stop words

1. **Finding Services**

**Services:**

**Search Engine**

* **Retrieve documents** with a certain word (Calls on setup for Search)
* **Retrieve documents** that contain a # of words(calls on Multi-Word Query with these words as the parameter , Multi word may be combined into one class for convenience)
* **Return Error** if user enters a stop word

**CLI**

* **Takes a string of commands,** and utilizes services of other classes for members

**Setup**

* **Scans documents** for words, and creates a list of groups with documents that contain a certain word
* **Scans stoplist** to avoid making this list
* **Takes a group of words** and finds the buckets from setup with that group of words & **Scans said buckets** for similarities, and returns those as a new query not part of current list
* **Constructors:**
* **Search Engine: No**
* **CLI: Yes**
* **Setup: Yes**

1. **Finding Inheritance**

Some of the things that I noticed about the setup was that a lot of the computation can be mostly done by the setup class, and that the CLI is only necessary for processing commands, and that the Search engine only has to retrieve the list of documents, and could possibly be combined with the Setup class. I have also made the design decision of doing all of the multi word processing within the setup class, as it is the most structurally efficient.

Out of the three classes that I have found, I think that the best class if I did have a superclass would be the setup, as it can perform the most services, and theoretically could complete the services of search engine as well.

1. **Finding Delegation**

Some of the delegation that I found throughout the document was the multi-query search, which might be easier as a subclass, but I am not sure yet what would be the best option for this assignment.

1. **Filtering**