# Approach to the problems

1. Tagging:

The tagging of the videos has been done using Alchemy API. More details about the API could be found at <http://alchemyapi.com> (api\_key.txt file consists the AlchemyAPI key)

Modules involved:

pyAlchemy package/

AlchemyStructure.py

ProcessAlchemy.py

alchemyapi.py

VideoTagging package/

taggingVideos.py

Utility package/

readCodeDataset.py

textProcessing.py

The tagging is mainly done by extracting the named entities mentioned in the text of the Youtube videos. Both the title and the description of the videos have been considered. The choice of only named entities over concepts, keywords and categories gave better results (just telling through manual checks, didn’t employ any objective evaluation measure like precision and recall). Moreover, named entities give a clean segregation.

1. Video Recommendation

Modules involved:

VideoRecommendation package/

videoRecommendation.py

nltk package/

Utility package/

textProcessing.py

The recommendation algorithm is based on tags that have been extracted for each video.

* 1. Two Hash tables are maintained, one of them stores the videos by their system generated ids (uuid) and the other one stores the list of video ids by the tag associated with them.
  2. For each video the associated tags are taken, then a lookup is made in the Hash table for finding the list of videos associated with each of the tags.
  3. Each video also has a termVector associated with it, which is a set of tokens after removing the English stopwords and stemming the remaining words. Porter stemmer is used for stemming. The text is tokenized at the blank spaces after removing the digits and a list of special characters.
  4. After fetching the list of videos associated with the tags of the current video, the intersection of the term vectors between the current video and the related videos are calculated. They are then sorted based on the number of intersections. The more the number of intersections, the more similar the related video is with the current video. The related videos are then ranked according to their similarities. Top 3 videos are shown based on their similarity.