Candidate Evaluation Assignment for Backend Developer

You have two weeks to complete this assignment. Please read the assignment and follow instructions to develop your own application as required:

Introduction:

You will download 4 ebooks mentioned below and parse the text. You will then perform text analytics tasks on your data and store the results in your DB. You will then build an API to serve those results on request.

In Summary, the assignment comprises of:

1. Parsing text
2. Applying text analytics to the data
3. Storage of data
4. Serving the results over an API

Description:

- For the purpose of this assignment you can choose a language of your choice out of the three:

1. Python
2. Java
3. Scala

- You will download and use the following ebooks:

Please download from the "Plain Text UTF-8" Link

1. The Notebooks of Leonardo Da Vinci : http://www.gutenberg.org/ebooks/5000
2. The Outline of Science, Vol. 1 (of 4) by J. Arthur Thomson : http://www.gutenberg.org/ebooks/20417
3. Ulysses by James Joyce : http://www.gutenberg.org/ebooks/4300
4. The Picture of Dorian Gray by Oscar Wilde : http://www.gutenberg.org/ebooks/174

- Text Analytics:

1. **Word Processing:** First you will need to read the files and process the text using NLP rules. You will be required to cleanse the text of any undesired artifacts and produce a list of words. There should be **no stop words** in this list. Stop words are basically the most common words used that are important to sentence construction but add little to the core features of a sentence. For Example: a, the, from, to, of, be, etc. You can

either formulate your own list of stop words or use a predefined list. Please state your choice clearly.

2. **Stemmed / Lemmatized Word Count**: Next you will produce a list of words after stemming or lemmatizing them to their base form (Example: go, going, gone all reduced to go). The purpose here is to reduce words to their base form and count the occurrence of each base form in a document. You are free to choose between stemming or lemmatization techniques but must state your case and methodology for your choice. You will compute a list and total number of **UNIQUE** base words. You can also use any of the NLP libraries mentioned later, for this part. Note that for part 3 you will only use these reduced base word forms. You can decide if you want to do the same for part 4 or not. For help and reference:

   https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

3. **Parts of Speech (PoS) Count:** (Nouns, Verbs only) Next you will find out the total counts of Nouns and Verbs from these **BASE** words in each document. This means that your application will (A) Find and store all the Nouns and Verbs in a document and then (B) Find and Store the total number of Nouns and Verbs in each document. Example- A: (Word : "Apple", Count : "7") - B: (Nouns : "54463", Document : "The Notebooks of Leonardo Da Vinci"), etc. We recommend using NLTK (https://www.nltk.org/), Stanford CoreNLP (https://nlp.stanford.edu/software/index.shtml) or CLIPS-Patterns (https://www.clips.uantwerpen.be/pattern) though you are free to choose any other library or write your own implementation for Noun - Verb extraction.
4. **Document Similarity / Difference *(Bonus)*:** In this section, you will provide a Document similarity or difference score for each pair of the mentioned documents. You are welcome to use the statistics you have derived earlier in your evaluation or choose any other approach you may think is more appropriate. You must clearly detail your choices and your approach for this part. Since this is a rather subjective problem, It would be wise to look at comparative approaches and decide on the one you think is the most useful, and make a case for it.

- Storage:

You must store all computed metrics from the last section in a database. You can choose between:

1. mySQL
2. Mongo DB.

- Application Programming Interface:

Note: *Please design, specify and Document all the parameters and specifications for the mentioned API*

You will setup an API to serve the result of the computed stats from the DB to a user on request:

1. Base Words:
    1. User can send a request for base words and is returned a list of all unique base words and their respective counts
    2. User can send a base word or a lemma of the base word (Ex: go or going) and, is served the count of the particular query in return
2. Noun / Verb Count: User can send the name/id of a document and is served the Noun / Verb count as computed above
3. If the user does not specify a particular document, he is served the statistics for all the documents
4. If the user specifies an ID/Keyword (such as ALL) for documents then the stats are combined for all documents and returned (Word : "Where", Count : 12841, Document : ALL, means the count was computed across all four documents) This should work for all computed statistics
5. **Document Similarity / Difference *(Bonus)***: Use can simply send a query by selecting any two documents and is returned a score or a representation of the similarity / difference between the two documents. Alternatively user can send a query for all documents and is returned comprehensive results of all possible pairs of documents compared to each other and their similarity / difference metric

Evaluation:

Please Note that you are the designer of the application. This means that **other than what is specified**, you are free to design your APIs / use appropriate libraries for each task or you could simply write your own implementation. You are also free to decide which metrics and computational strategies are best for any task (Example: what is the metric for document difference and how to compute it). You will of course be evaluated for how well your methodology produces the desired results. Your code will be evaluated not only on how many tasks you complete but more importantly on how well does your application perform algorithmically and functionally. Therefore, **please make sure to comment your code / provide additional documentation explaining your choices and the reason you think they are the right way to do the job.**

Submission:

You can submit your assignment in one of the following two ways:

1. Commit your assignment on your Github account and share the repository
2. Bundle your submission as a Zip/RaR file and send it via email

Please feel free to contact us if you have any queries. Best of Luck !!!