The Floow: Platform Engineer Challenge

Introduction

The Floow uses Java and several other technologies to process data as part of their Telematics Platform. To ensure reliability and performance, many components of the service must operate in a distributed fashion to balance load and maintain continuous operation. This test is designed to assess your ability to create a system that achieves these objectives on a smaller scale.

The Challenge

The functional goal of the challenge is to analyse a file with a large body of text and to count unique words so that the most common and least common words can be identified. Your solution should also provide a means of viewing the results.

The technical goal of the challenge is to create a system that distributes the workload and scales easily. Your solution should demonstrate that *you are capable of engineering* a system, we therefore discourage the use of frameworks that will manage the distribution in its entirely[1].

You are required to produce a program that counts the words in a file and saves the counts to a MongoDB server. The program will need to support execution on multiple servers that communicate via a common means (e.g. a MongoDB collection) and work together to break down the workload.

The challenge should be capable of being executed as a JAR on a number of servers. Ensure that the program can be run using simple command line arguments as below.

For example:

```
java –Xmx8192m -jar challenge.jar –source dump.xml –mongo [hostname]:[port]
```

Objectives

Mandatory Goals:

- ❏ Produce the solution using a git repository and demonstrate proficiency with using git to develop a software project
- ❏ Create a single program to perform the task, that can be run using command line arguments
- ❏ Distributes the workload to run on any number of servers
- ❏ A user should be able to view/query the results (most and least common words) of the program
- ❏ Demonstrate an ability to use and understand MongoDB

[1] For example: Apache Spark

Desirable Goals:

- ❏ The solution should recover gracefully from the failure of a server
- ❏ Report on statistically interesting words (in addition to the least and most common words)
- ❏ Notes considering trade offs between efficiency and accuracy of the solution

Test Data Set

While you can test and develop using any data set, this challenge will be assessed using the following (large) dataset which can be downloaded from dumps.wikimedia.org/enwiki/latest/

- enwiki-latest-pages-articles-multistream.xml.bz2

Please Note: *These links change from time-to-time and this one may no longer be the most up-to-date.*

If it is difficult to acquire we can provide a copy of this dataset – Wikipedia dumps can be unstable at times.

You will not be expected to account for the fact that it is XML based, no extra credit will be given for doing so.

Submission

Your submission must include:

- ❏ Complete source code hosted in a git repository such as GitHub[2] or Bitbucket[3]
- ❏ An executable JAR which can be run using the command defined in "The Challenge" section above. You may provide your jar as part of the repository or separately (e.g. zipped attachment to your email).
- ❏ A README file within the repository, that includes:
  - ❏ instructions required to successfully run your solution and describing how to view the final results (the word counts)
  - ❏ A description of the technologies/architecture used in your solution

[2] GitHub lets you have an unlimited number of free repositories

[3] Bitbucket lets you have an unlimited number of free and private (if shared with no more than 5 people) repositories.