Sparkfly Golang Code Challenge Project

Summary

The Sparkfly code challenge project consists of two small exercises meant to test your understanding of Golang's parallelism features as well as your understanding of common Golang base library functions and interfaces.

These challenges are meant to be completed in a couple of hours. If you find yourself struggling, or have questions about the requirements, please feel free to e-mail us.

Writing tests for these projects is not required, but you may find it helpful to do so.

1) **Parallel Uniqueness Checker:** We have generated a number of CSV files containing random strings that should be guaranteed to unique (e.g., X59J) within themselves and among each other. Write code that can take the names of these files and verify the code uniqueness across all CSV files in parallel. Bonus points if you can make it immediately stop itself once it has found a duplicate code. Example CSV files can be found in the attachments of this email.
2) **Basic Byte Compressor:** We receive large (20MB+) text files that have to be stored in S3 for safe keeping. To minimize costs, we would like to store them in a compressed format. To further minimize costs, we would like to offload this process onto low-memory hardware. We get these files regularly and need the software that processes them to be expedient. For simplicity, we have decided to use the GZIP compression format as it offers the balance between speed/compression that we need. Please write code that takes uncompressed input and writes compressed output. The interface requirements are:
    a. The upload manager to S3 takes an **io.Reader** as its argument (output from your code)
    b. The uncompressed data is provided to your code as an **io.ReadCloser** (input to your code)

    Here is an example of the interface your code should satisfy:

```
// YourSolution is an io.Reader that provides compressed bytes
type YourSolution interface {
	io.Reader
}

// NewYourSolution takes a source of uncompressed bytes
func NewYourSolution(rc io.ReadCloser) *YourSolution {
	/*instantiation code here */
}
```

Submitting

You may submit the solutions to us via ZIP file (e-mail) or public Github repository. README is appreciated, though not required.