Freshworks – Backend Assignment

Build a file-based key-value data store that supports the basic CRD (create, read, and delete) operations. This data store is meant to be used as a local storage for one single process on one laptop. The data store must be exposed as a library to clients that can instantiate a class and work with the data store.

The data store will support the following **functional requirements**.

1. It can be initialized using an optional file path. If one is not provided, it will reliably create itself in a reasonable location on the laptop.
2. A new key-value pair can be added to the data store using the Create operation. The key is always a string - capped at 32chars. The value is always a JSON object - capped at 16KB.
3. If Create is invoked for an existing key, an appropriate error must be returned.
4. A Read operation on a key can be performed by providing the key, and receiving the value in response, as a JSON object.
5. A Delete operation can be performed by providing the key.
6. Every key supports setting a Time-To-Live property when it is created. This property is optional. If provided, it will be evaluated as an integer defining the number of seconds the key must be retained in the data store. Once the Time-To-Live for a key has expired, the key will no longer be available for Read or Delete operations.
7. Appropriate error responses must always be returned to a client if it uses the data store in unexpected ways or breaches any limits.

The data store will also support the following **non-functional requirements**.

1. The size of the file storing data must never exceed 1GB.
2. More than one client process cannot be allowed to use the same file as a data store at any given time.
3. A client process is allowed to access the data store using multiple threads, if it desires to. The data store must therefore be thread-safe.
4. The client will bear as little memory costs as possible to use this data store, while deriving maximum performance with respect to response times for accessing the data store.

Languages:

Ideally, we would not restrict you from working on a language of your choice. However, it would be preferable if you stick with one of these -

- NodeJS
- Java

- Python
- GoLang
- Ruby
- C/C++

Submission:

Submit a link to the source code, ideally committed to github.

Code accompanied by thorough unit tests is typically a mark of quality work.

Ideally, your data store will work on most operating systems. If this is not the case, please specify which OSes are supported.