Final Project

Build a document DB that fulfills that following requirements:

- Reads are the majority of transactions (from any node in a cluster)
- Writes are done only rarely (only through a connection to the database controller/connection manager).
- Has to have fast reads (e.g. caching in memory).
- Document based database (json objects).
- Each document type has a json object schema and each object schema should belong to a database schema.
- A document has an ID which should be unique and indexed in an efficient manner.
- Create indexes on a single json property.
- (Optionally) support multi-property indexes (static indexes).
- Data and Schema changes including Index initiation and creation will be done on the controller but Data and schema and indexes will be replicated to nodes so that searches for reads are done locally in a node.
- Has the ability to scale the cluster horizontally (data should replicate to new nodes up to 3 nodes or more), the client will ask the database controller/connection manager for a connection (to a specific database schema) and this should connect it to one of the load balancing nodes for the duration of the client connection (e.g. docker network or different ports for the nodes or virtual machines).
- Allow for database schema to be exported and imported (from controller).
- Default administrator to use Controller to manage roles of users and administrators (login name, password and role as user or administrator), default administrator has to change his password the first time.
- Don't use indexing libraries such as Apache solr.
- Controller is a bottleneck and single point of failure, typically it should be redundant but this is not required at this stage.
- create a demo application that is either a command line application or a web application that utilizes the database using its own schema (e.g. email application, catalog application…etc).

Hints: when the schema/data/indexes are being updated by the controller, and there are multiple readers reading at the same time , they should be using the older version of the schema/data until it is updated. The race conditions have to be managed in the most efficient way.

Write a report that includes:

- How you implemented the DB,
- The data structures used,
- Multithreading the locks,
- Scalability/ Consistency issues in the DB
- Security issues in the DB
- The protocol you implemented between the client and the server.
- Defending your code against the Clean Code principles (Uncle Bob).
- Defending your code against "Effective Java" Items (Jushua Bloch)
- Defending your code against the SOLID principles
- Design Patterns
- DevOps practices