Matthew Spencer

SNHU CS-499

Milestone 4 Narrative

Artifact Description

Computer Science 340 introduces advanced programming concepts to the student. In essence, the class is based around the idea of incorporating a database into application development. The artifact created in the class is an application which provides a RESTful interface for updating and maintaining a mongoDB as well as an application with a user interface which allows the user to perform basic crud operations on the database.

Justification for inclusion

Category three covers database development. This artifact is the strong enhancement of a spring boot application using a MongoDB back end for the purposes of performing CRUD operations to a set stock data. The original class was focused primarily on learning MongoDB operations within the database itself. In practical terms, this works well for a DBA, but is not as useful to a developer or DevOps engineer.

MongoDB provides very strong Java libraries which enable a developer to create the MongoDB database and collections on the fly. In addition to these, java is able to incorporate aggregations in a smooth and relatively painless manner. While MongoDB provides a native RESTful interface, building up a Java front end application can provide a more user-friendly experience without sacrificing any capability.

Coverage of Course Objectives

Changes to this artifact enable the application to create the MongoDB collections on-the-fly rather than depend on them being created in a database prior to the deployment of the application. By adding the

appropriate annotations, indexes and default sorting can be applied in such a way that the Java class definitions contain everything needed to spin up a solid MongoDB collection. Being able to create the collections as needed saves a considerable amount of configuration time. In addition, it ensures the database contains only the information needed by the Java application rather utilizing a great deal of unnecessary storage holding unneeded definitions.

Storage, manipulation, and access of data in the MongoDB is all done programmatically through Java code. The MongoDB libraries provides straightforward commands for performing all crud operations. Additionally, because the crud operations are tied closely to the Java objects, compound queries for saving or reading data is unnecessary. For example, if a person has an address, it is not necessary to perform two operations to pull the person and the address, nor to save them. The entire tree of objects can be pulled or pushed via single read or save command.

Security within the database is a concern. In one particular instance, this application leverages username and password for access. Passwords are encrypted within the Java objects and stored in the database as encrypted text. MongoDB supports configuration role-based access to both databases and collections. Where the data, can be configured on the fly, the role-based access to the database itself must be configured separately at the DB host level. For sake of simplicity and ease of reproduction, this step has been bypassed allowing open access. Configuring the authentication in the Spring Boot application is a matter of adding the appropriate properties values to match those set on the MongoDB server itself.

Even with a secure database, it is possible to allow unauthorized users to make changes to the database through inadequate application security practices. Adding authentication verification and restricting views ensures that only authorized users can perform actions such as writing to the database. Less restrictive access is applied to read-only functionality within the application. Having authentication in general prevents unauthorized users from having access to the data ingeneral.

Reflection

All in all, I see this milestone as the foundation of, and in some respects the extension of the previous milestone. As stated in my previous reflection, it is impossible to create a database driven application without extensive use of data structures and algorithms. However, using the same artifact for both milestones gave me an opportunity to dig further into every aspect of the application and create something more functional and user-friendly.

Adding a security layer to the application necessitated creation of new collections for roles and users in the database. Additionally, new functionality within the web application allows an administrator to change access levels of existing users without having to touch the back-end database.

I made one conscious decision with regard to security in the application. While the website side of the application requires authentication for access, the REST interface requires authentication only for write actions. I decided that it would be interesting to create a rest interface that is open to general use which allows a user to access the data freely leveraging it for their own implementations.

Overall, this has been a fun experience. To me, having a question, “I wonder if I can do ____”, and then doing the research to make it happen is the most fun I have in software development. It’s fun to take an iterative approach, testing things out in parts and pieces, and then seeing it all come together in the final product.