4-2 Milestone Three: Enhancement Two – Algorithms and Data Structure

Larry McCoy

Prof. Brooke

CS-499-T3724 – Computer Science Capstone

Southern New Hampshire University

30 Jan 2022

The first aspect of starting the process of converting the code base from Python to JavaScript (JS) was to conduct a thorough review of the existing code from the previous Milestone assignment. That combined with the generated RTVM created, an assessment of just how to develop the new JS structure was used in determining how the files would be used to execute all the different functionality needed to accomplish the desired program outcome. One of the toughest challenges I had was limited familiarity with the JS language in which I relied heavily on the website <https://www.javatpoint.com/javascript-tutorial> to help provide insight on how to convert the existing Python syntax into JS format. The two main enhancements provided for this module were to convert into the JS language which would utilize an SQLite database meaning the data structure was going to need changes in order to operate within a relational database. The second was to incorporate unit testing in order to meet the standards associated with static code coverage.

To meet the first rubric measurement, demonstrate innovative skills and techniques for implementing design solutions, I determined that there was a need for the main system function to set up the program for basic operations. The artifacts included to show this are the system.js, main-menu-screen.js, animal-screen.js, and the animal-service.js files. Additionally, I knew there needed to be some tool to help accomplish the unit tests as well as a method to report the results. Through some investigation and inquiries form co-workers I was able to identify the Mocha test tool and C8 reporter were compatible and very successful when using the Node.JS platform. I also had to construct a test model to use memory to simulate a database for testing the animal-service code, this is still a work in progress as I continue to work on the functionality code base. An index.html file has been included with the artifact documents to show the current unit test results.

To meet the needs for the update of the database from MongoDB to SQLite, a shift from storing data in flat-file documents needed to be made. Data structure format for the various class object and stored values shifted to include Integers, strings, float, Boolean, and double data types. To allow for these to be used globally, model files were created to handle all the class structures so they could be called via the primary service modules. This is one aspect to accomplish the second measurement in the rubric for considering the data structure template. The second aspect is the use of algorithms, since the initial program did not utilize any algorithms for functional execution there also are none being used here. I did, however, determine that a bulk/batch creation algorithm is going to be used when seeding the database upon initial launch of the program. The previous system held over 10,000 records which was looking to be an excessive amount of time to populate into the database. This algorithm will help to read them in and store much quicker.

The approach taken to demonstrate measurement three in the rubric is by conducting a systematic approach to the code development. Starting with the main menu selection and slowly implementing each desired function, I was able to identify potential overflow areas when it came to displaying the main menu. I continued that process through the implementation of the animal-service path associated with the CRUD operations. This has taken a little longer than expected, especially with debugging flaws that were introduced with each new coded segment. The process of building unit tests as the program was being written assisted greatly in finding many of the injected issues and allowed for quicker resolution during the debug process.

Overall, things are progressing rather well with still quite a bit of work left to accomplish. We currently have the Create and Read functionality fully operational in the code with appropriate unit testing coverage. The Update and Delete is not far behind leaving only the filtering functionality to accomplish. As it was difficult to test without it, most of the database work is completed as well. This will help the project to come to completion much quicker. All code artifacts are included showing the progress to date on the main functionality.