Milestone 4 Enhancement

William P. Giittinger

william.giittinger@snhu.edu

Southern New Hampshire University

**Zoo Authentication Website Database Artifact**

This artifact contains the creation of the backend database (MongoDB), created through Atlas and hosted by Amazon Web Services (AWS), for the sample zoo website. The database was created from scratch and modeled after the files and data already in use on the website. The primary goal was to take the data from these files and transfer it to a document (object) oriented database for data management. Mongoose was used for data modeling on the server to ensure proper data structure and information. The database itself only contains two collections, one for the zookeepers (users) and one for the animals. Once the database was up and running, the application needed to be modified to connect and authenticate with the database. Then the old file-based methods and functions needed to be refactored to use the database rather than the server-stored files. Ultimately, this allows for easier data management in the future including storing large data sets, easier data modeling, security improvements, easy indexing, etc., employing all the benefits of using a MongoDB.

The creation of the database is a cornerstone for the application. Of course, data can be stored in application memory, but this inefficient and comes with a host of issues, perhaps most notably, including the lack of support for multiple users. Mongo also stores its data in documents, which are the same structure as JavaScript objects, which makes them much easier to work with. One can add, delete, search, filter, update, etc. documents through Mongo. While not all of these interactions are coded into the application itself, it is possible to still access these functions through other Mongo utilities such as Compass, which also provides data visualization. This artifact displays a variety of key software development skills. Firstly, the ability to setup and connect to a database and then manage that connection in the application. This step requires setting hidden environment variables (URI paths including host information, username, password), connection parameters, and handling different types of application terminations which also need to terminate the current database connections. Next it showcases that I can interact with the database and utilize the data stored on it in a variety of ways (CRUD functionality). Lastly, due to the development lifecycle of this application, it also shows that I can refactor existing functions to use the data from a different source.

I believe that the planned objectives for this artifact were met. Specifically, manually setting up a brand-new database from scratch and utilize it for data management for the web application. This shows my capacity to work on both front-end and back-end portions of a website in a full-stack manner. Due to the order of artifact creation, with the database being the final artifact created, there were additional challenges when implementing the server into an application that was created without its use to begin with. However, I believe this goes to show that I am capable of refactoring code to accommodate changes in design and function. With this artifact in place, there are no updates to the outcome-coverage plans assuming positive feedback for this submission. If there are issues, those will be addressed, but if not, a final polishing-pass will be done on the application for any major usability issues.

The process for creating and implementing the database for this web application had both easy steps and much harder steps. Firstly, I have never created a database from scratch in this way before. I have worked with local databases on my own computer, with heavy guidance from the courses that I was in. However, this database is hosted by AWS and can be remotely connected to from other client devices. Setting up the environment variables and connection methods for connecting to the remote database were not overly difficult but were brand new to me. Next, working with mongoose for modelling the data was an interesting challenge. I have worked with mongoose in the past, but again with plenty of guidance. Setting up the schema from scratch and learning how each part of the tool worked was challenging and eye opening. Of course, there were plenty of challenges refactoring most of the work I had already done to convert it to use data coming in from the database rather than reading from files. To make that easier, I used the same general object structure as before, but it was a fair amount of work to get the methods to work properly. Not only that, but the methods are essentially duplicated as I left the binary tree structure on the application itself. One of the most unexpected challenges was reworking the configuration for the passport module, which was essentially entirely scrapped and redone. I think this goes to show that with so many moving pieces, unexpected challenges are bound to arise. However, I tried to take the conversion process one step at a time, and ultimately I was able to setup my first self-created backend with MongoDB.