Ben Robbins

09/29/2024 – Updated 10/19/2024

Milestone Three

**Enhancement Two: Algorithms & Data Structure**

For my enhancements in algorithms and data structures, I continued to use my Travlr Getaways full-stack web application project as my artifact. This full-stack application project was initially completed using the MEAN stack (MongoDB, Express, Angular, and Node) for CS-465 Full Stack Development. The original project scope was to convert one static html page into a dynamic webpage and to create an administrative single-page application. Both the client-side and back-end of the application were developed for these requirements. I selected this artifact for this enhancement because only one page of the application was originally developed. The remaining undeveloped pages provided multiple opportunities to demonstrate my skills and ability to meet the course outcomes by increasing the scale and complexity of the project.

To implement these enhancements, I planned to design and develop the APIs needed for the newly created pages of the application to access and dynamically display data stored in the back end of the application. The APIs I planned to develop would consist of methods to retrieve data (i.e., GET requests), add data (i.e., POST requests), and update existing data (i.e., PUT requests). Additionally, I planned to implement security measures to protect the POST and PUT requests from unauthenticated users by using JSON Web Tokens. These enhancements will demonstrate my skills and abilities in algorithms and data structures and security best practices.

To implement the APIs, I created a controller for the rooms, meals, and news pages. These APIs connected the front-end to the back end of the application. In each controller, I created four methods; a GET method for all documents, a GET method for a single document, a PUT method, and a POST method. Additionally, I implemented error handling to account for any undefined behavior when attempting to retrieve the data. The methods were exported from the controller to be accessible to the routes. I updated the existing index.js route file to import the newly created API controllers and define the URLs for the APIs. To utilize the APIs in the application, I then had to update the controllers for the different pages of the application. Rather than reading the data from a static JSON file, I updated the controller’s functions use the API endpoint for GET requests and use the returned data in the dynamic display.

In these enhancements, I have successfully designed and developed functional and efficient APIs that support the overall goal of transitioning this static website into a series of dynamic webpages. Not only do these enhancements demonstrate my ability to utilize algorithmic principles and best practices, but I have also demonstrated my ability to develop an application using a security mindset. To maintain the integrity and security of the data stored in the external MongoDB database, I implemented controls to protect the APIs that can alter or add data using JSON Web Tokens. The JSON Web Token acts as an authentication security layer for the PUT and POST APIs. Each API function requires the user’s JSON Web Token to be verified before any data is altered to prevent an unauthorized user from altering the data.

Reflecting on my development process, one challenge I faced was maintaining my organization and file management. As I have increased the size and complexity of this full-stack application, I found it much easier to become disorganized when switching between so many files. As I developed the APIs, if I had to take a break or even just remembered something I had previously forgotten to do, I could quickly lose track of my progress and make mistakes. While I was able to debug the self-imposed problems that I created, it decreased the efficiency of my development process. To counteract this, I used the unique functionality of VS Code to my advantage. I split my screen to display two files at a time so I could focus on the controllers I was creating as well as the route file index.js. While it may seem simple, a simple solution is often the best solution.