The artifact that I selected for the first round of enhancements is the travlr getaways full stack web development application. This was originally created and worked on in cs 465. The original program was a server rendered website that was built using node.js, express, and handlebars to display a selection of static vacation packages. It had basic routing, some simple controllers, and a json file for the trip data. It did not have any dynamic data storage, authentication, API structure, or modern security considerations. I have chosen to enhance this project into a more realistic release ready application by reorganizing the structure, implementing a MongoDB database, adding secure user authentication, and building a complete rest API with protected crud operations.

I chose this for the eportfolio because it will showcase my growth in software engineering, architectural decision making, database integration, and secure API design. The enhancements required me to restructure the application, where the server rendered pages and API functionality would be separated. I also added a dedicated app_api layer that has models, controllers, routes, and middleware, and I added a persistent MongoDB database using mongoose. I also added a more complete user model that has hashed passwords, a jwt based authentication system, an auth controller, and access control middleware which protects the crud operations on trips. I also enhanced the Angular front end by modifying the trip data so that it included JWT tokens in HTTP headers for protected routes. These enhancements all showcase skills in backend development, restful design, authentication workflows, and frontend to backend communication. It now reflects a more modern, secure, and database driven full stack application instead of a simple static design.

From my planning I wanted to demonstrate my ability to use professional software engineering tools, practices, and patterns, and develop a security mindset that anticipates vulnerabilities and implements appropriate safeguards. I believe I have met both outcomes. I am using modern tools like express, mongoose, jwt, passport, angular, and applied more industry standard practices like MVC separation, API route modularization, password hashing, token-based authentication, input validation, and more consistent error handling. The security has also been enhanced by moving away from static insecure data structures. I believe I met my initial goals and even did more than originally intended when I improved security, and my engineering practices, but also improving the clarity and long-term maintainability of the system.

While working on this I did learn a bit about taking something simply made for academic purposes and evolving it into something that is closer to real world software. Some of this learning came from refactoring my own earlier iterations of my work. I was able to see small flaws and could see how the code would become harder to maintain

without proper structure, which reinforced how crucial modular design is. I also learned a bit as I explored how to make it more secure, with hashing, token generation, and token verification. I would say that the biggest challenge was taking the existing code and using new architecture, like converting the static Json data into a database model and making sure the older handlebars-based screens still work when new API is added. Some issues also came up when debugging middleware order, token formatting and the database validation, which caused a slight slowdown as I reconsidered the data flow. Overall, this allowed me to showcase the growth I have made since taking the course but also allowed me to continue to grow with code organization, debugging, and boosted my confidence levels when working with full stack tools.