Erik Larsen

Southern New Hampshire University

Database Narrative

This artifact is a recreation of the RESTful API that I created in the last class I have taken, CS340. It works by accepting HTTP requests and returning the requested information from the database. A GET request can get the info for any stock from the ticker, a POST request creates a stock in the database, a PUT request updates, and a DELETE request removes the info from the database. This all mirrors the python API that was created previously, but now I am using the MEAN stack with MongoDB, Express, and Node.js.

I selected this item to show that I can use the same principles in different languages, as well as show some improvement. The first creation of this API I did not always use proper status codes or paths for the requests. This has been improved on the current artifact. Also, by changing languages, I showed that I can learn how to obtain the same functionality through different languages and understand documentation for learning more packages etc.

Currently this shows my ability to use well founded skills in computer science by using multiple languages and packages etc. This artifact also shows an eye towards security with the inclusion of private routes and other security features. Private routes are used to only allow authorized users access to Create, Update, and Delete functions. This prevents general users from being able to manipulate data, while authorized users can update the data as needed. Other security upgrades included the addition of some dependencies that sanitized data to help prevent noSQL injections. We also prevented rapid sending of requests from a single IP to help prevent brute force attacks. A normal user wouldn't need a large amount of requests in a 10 minute period, so these requests are rate limited for additional security.

When I was enhancing this product, I have learned a bit more about how RESTful APIs work. I felt lost while working on the original project and having this video tutorial to help describe everything was a great help to show how things work. I also learned more about proper routing and status codes to make sure everything follows the standard protocol. This makes my API more useful and easier for others to work with as it follows their standard naming protocol etc. I had a few challenges with the scope of the project. There is a lot that goes into this API and learning about middleware and controllers was a bit interesting. I am glad I decided to allocate the most time to this portion of the project thanks to this. The security aspect of feedback was incorporated by adding the additional dependencies to prevent injection. The private methods also help with ensuring the database is secure.