Category One Narrative

Briefly describe the artifact. What is it? When was it created?

This artifact is a course catalog application. In its original form, it was a C++ console app that read input from a file and loaded courses into a binary search tree to be queried. This version was created in a Data Structures & Algorithms course I took last year. In its new form, it is a web application making use of the Dash framework with Python to provide an HTML interface with which users can interact.

Justify the inclusion of the artifact in your ePortfolio.

This artifact was one that I was initially proud of in its C++ form. However, for something like a course catalog, the simple functionality of the binary search tree felt a bit lackluster. There was important functionality that I wanted to add and that I thought would be better handled by a database in a web application. This artifact shows my ability to take a piece of code written in one language with a particular structural implementation and convert it to something that's arguably more efficient and effective by making use of my skills as a well-rounded developer. My competence in both C++ and Python allows me to pick out the core components of the application logic and convert between the two languages. This artifact also proves my ability to create an intuitive HTML interface using the Dash framework, showcasing my UI skills. Finally, this application was built essentially from scratch. I had absolutely no Python code for this particular application before beginning to write it, so this artifact demonstrates my ability to take a process through the development cycle, redesigning, implementing, and testing.

Did you meet the course objectives you planned to meet with this enhancement in Module One?

The objectives I originally intended to meet with this enhancement were:

Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals

Develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources.

The finished product clearly demonstrates the progress made in both of these categories. First, the original project was in C++; this new version is in Python. I realized that Python would provide a better platform to create an application of this nature. I was able to use several packages written for Python to achieve the exact functionality that I desired for the course catalog. Dash, Plotly, and PyMongo are a few examples. Furthermore, I was able to analyze the core components of the app in C++ and translate them into Python while adding new functionalities like the ability for students to log in, register for courses, and complete a degree audit.

In terms of security, the program requires users to sign in in order to register for courses. When creating accounts, user input is limited to a small number of characters to prevent buffer overflow. Additionally, input is validated and sanitized before being sent to interact with the Mongo database in order to minimize the likelihood of injection attacks. The original program had no input validation and no way of authenticating users, so drastic improvements in the software security have been made

Reflect on the process of enhancing and modifying the artifact.

I learned a lot enhancing this artifact. The main goal of the enhancement was the language conversion and the introduction of an HTML interface. I knew that I wanted to add some new functionality, but I wasn't yet sure what functions I wanted to create. Once I ultimately settled on student registration and degree audit functions, I thought it would be rather simple to create the logic and implement these things into the interface. I was definitely mistaken. The development simultaneously of logic in Python that will respond to feedback from the HTML interface can be very tedious. I got much more familiar with the documentation of Dash, Pandas, and Plotly trying to implement solutions that would bring about the desired functionality. Every line of code must be perfect from the input validation to the conditional statements in callback logic. I spent several hours just debugging this application after writing the initial code, and honestly perfecting the Dash HTML provided more challenges at times than the Python itself. Though I used the Dash library to create the interface, I spent a lot of time reading HTML documentation in order to get things how I wanted them to be. Also, throughout many of my projects during my time at SNHU, I have coded mainly in only one file; however, I noticed that during my scripting process, there were getting to be too many lines of code, so I chose to segment the program into various files to make the code more readable. I ultimately accomplished the goal of making it more understandable, but there were many instances in which errors in the main code were a result of issues in other files. Overall, this project gave me substantially more confidence in working with both Python and HTML, and I look forward to using these highly sought-after development skills in each of these languages to create more sophisticated web apps.