Kenyhacta Busby

CS 499

11/22/2024

Enhancement Two: Algorithms and Data Structure

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

The artifact is a security-focused project created in C++ as part of the CS 405 course. It involves creating uniform implementation guidelines, presenting a security policy guide for Green Pace, and ensuring that the development team adheres to best practices. It includes writing unit tests to identify vulnerabilities and addressing risks using 10 guiding security principles.

**Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in algorithms and data structure? How was the artifact improved?**

The ePortfolio was chosen to include this artifact because it shows how well I understand and use algorithms and data structures. It shows how I solved problems like allocating memory efficiently and handling input safely. Both are very important in modern software development. The validation logic, performance logging, and retry mechanism show that I can use data structures and algorithms to make programs more reliable and efficient. The retry mechanism shows how control flow and algorithmic problem-solving can be used to limit attempts at invalid input, making sure that errors are handled correctly and making the user experience better. Performance logging shows that I can use time-based metrics to analyze and rate how efficiently memory is allocated, giving me information about how well the program works. Input validation and SQL injection prevention also show how well I can use string manipulation and logical checks to make sure that programs run safely and reliably, keeping the system safe from malicious inputs while keeping it functional. Together, these improvements show that I can use effective algorithms and data structures in real-world situations. Retry limits were added, memory allocation logic was made better, and performance logging was added to make the artifact more efficient. Implementing a structured retry counter, checking user input for SQL injection risks, and adding time tracking to measure allocation performance were all needed for these improvements. These changes turned the artifact from a simple code into a complete program that shows how algorithms and data structures can be used in real life.

**Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?**

The planned enhancements to my artifact clearly demonstrate progress toward meeting the course outcomes. By implementing stronger input validation, retry limits, and detailed logging mechanisms, I showcased the ability to anticipate potential vulnerabilities, mitigate design flaws, and enhance system security, reflecting a proactive security mindset. Enhancing the artifact with retry logic, input validation, and optimized task queue management highlights my ability to design and implement computing solutions using algorithmic principles. These improvements also demonstrate structured problem-solving while emphasizing efficiency in execution. Additionally, transitioning from C++ to Python allowed me to leverage modern libraries and built-in data structures, like lists, to streamline development and improve maintainability. Integrating performance monitoring with the time module further underscores my ability to utilize innovative techniques to deliver robust solutions. By refining user messages and integrating clear, actionable logs, I ensured that the artifact supports effective communication for both technical and non-technical audiences, aligning with the goal of delivering accessible and user-friendly solutions. The enhancements also ensure that the artifact can be used in collaborative environments, enabling diverse teams to make informed decisions based on its clarity and security features.

Initially, the artifact did not fully meet the course outcomes planned in Module One because the focus on algorithms and data structures overlapped too heavily with security enhancements. To address this, I updated the artifact to place a stronger emphasis on distinct algorithmic principles, including input validation, memory allocation, and task management. These updates highlight the use of Python’s built-in tools and performance logs, which ensured alignment with the targeted course outcomes. Overall, the refinements successfully balance security, efficiency, and user-centric design while leveraging modern computing practices, fully meeting the intended objectives.

**Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?**

I realized how important it is to thoroughly document your code after modifying and enhancing the artifact. Documentation that is clear and to the point not only makes the code easier to read. It also makes it much easier to find and understand later, especially when making changes or fixing bugs. One challenge I faced was making sure that the updates made it clear which ones were focused on algorithms and data structures and which ones were focused on security. To make sure this fit with the second category outcomes, it needed to be carefully planned, and the artifact had to be reevaluated. Putting the improvements to the test to make sure they work while staying simple was another problem. Even with these problems, the process made me realize how important it is to code in an organized way and solve problems in a structured way.