**Software Engineering and Design Enhancement**

Brandon Cook

Southern New Hampshire University

CS 499: Computer Science Capstone

Dr. Pravin Bhandari

September 21, 2025

**Software Engineering and Design Enhancement**

The artifact used for this portion is an investment calculator console application that I developed during CS-210: Programming Languages. A user can enter in an initial investment amount, a monthly deposit, an annual interest rate, and a number of years for the investment to grow. The totals both with and without monthly deposits are calculated and displayed year by year in a formatted table for each outcome. It is a tool that can be used to compare the difference in growth when adding an additional monthly deposit to an investment.

My decision in using this artifact relied on a few factors. For one, this project is something that was written entirely by me. This was not, as is the case in other classes, a program that was written as a template that I then modified to fulfill a rubric outcome. All of the work was done from the ground up by me, and I was interested in focusing on something that I built myself, rather than something I added on to. Furthermore, enhancing this application further to incorporate additional features and set the framework for a shift in focus into an actual financial planning app would rely on several technical skills that were spread across several competencies. Adding in file handling, refactoring code to have a more intuitive user interface, adding additional security measures to prevent the system from being broken or unsafely crashed, and creating a set of unit tests using the Google Testing framework demonstrates a variety of skills that are useful in real world practice. It shows adaptability and flexibility.

The course outcomes I planned to meet with this first enhancement are as follows:

* Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.
* Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution while managing the trade-offs involved in design choices.
* 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.

I have created a thorough overview in a code review outlining the plan and scope of the enhancement project, I have adapted the program to write to a file that can be later imported and tested with automated unit tests to display problem-solving abilities and demonstrate innovative techniques and tool usage, and I have developed security measures to better protect the system from improper and malicious usage. I do plan to continue working on this before the final submission of this primary enhancement to my ePortfolio. There is not much left to do – I need to finish implementing the unit tests. I additionally want to refactor the current import and exporting system slightly, to allow users to name the files they want to write to and read from, so that it is not only writing to the “investment\_report.csv” file every time.

When I was creating the unit tests, I modeled the frame loosely on an assignment I had completed during CS 405: Secure Coding, where I had to add a few unit tests to a template involving operations performed on a collection. The tests used here are vastly different, but using that as a reference point helped refamiliarize myself with Google Tests and have a decent head start on the work. I did not begin to implement these tests until the end of the project, focusing on refactoring for the initial bulk of the work. I implemented a shell for the Export and Import functions following splitting up the main function from UserInterface, which is where I began to run into problems. Exports worked fine, but when reimporting, there were several issues. The biggest realization was that, in the original program, I was performing calculations and printing the output to the screen at the same time. The year-over-year data was not being stored in any way, only the end results, using a method that incremented the total with and without deposits over a loop for each.

I eventually implemented vectors for each item, the yearly balances and the interest gained each year, for both with and without the monthly deposits. This allowed all of that information to be written to a file, which could successfully be imported and displayed. At first, I ran into an issue following this where output was being displayed multiple times, but I realized that I was not clearing the vectors after each import, so it was doubling the values. Resolving this, I worked on adding some limitations and reformatting the output to be cleaner and easier to read. I began by implementing some realistic limitations, such as $10,000,000 as a maximum initial investment, $10,000 as a maximum monthly deposit, 100% as a maximum interest, and 30 years as a maximum amount of time. I implemented most of these limitations and measures into the if statements that assign them, in an effort to minimize the lines of code. I also tried to keep it as easy to read as possible, by implementing constants for the maximum numbers and using those in the statements, rather than placing the number directly. In future iterations, this will reduce confusion as to what the number is meant to represent.

In summary, I feel like I learned a lot and had a chance to develop specific skills that I want to focus on further. Unit testing is a skill that I want to exercise and work more with. In future projects, I would like to begin by creating the unit tests first, which will help to guide my progress and act as a reference for functionality that needs to be in place.