# CS 499 Milestone Two Narrative

## Artifact Description

The artifact I selected for this milestone is my C++ Reverse Software Engineering project, originally created during the Fall 2024 semester for CS-410. This console-based application simulates basic user interaction and access control through a menu-driven interface. Users log in with credentials, navigate options, and perform placeholder actions. Its initial design included hardcoded authentication and minimal input validation.

## Justification for Inclusion

I chose this artifact because it highlights core skills in software design and engineering, especially in the context of secure coding and refactoring legacy code. By enhancing this project, I showcase my ability to implement modern security practices, such as password hashing, input validation, and modular class structures, and improve maintainability and scalability. The refined authentication module, clear separation of concerns, and robust error handling demonstrate practical applications of SOLID principles and secure development standards, making this a strong portfolio piece.

## Course Outcomes Reflection

In Module One, I planned to meet outcomes related to designing secure computing solutions and developing a security mindset. Through these enhancements, I have achieved strong competency in those areas by implementing password hashing, input sanitation, and structured logging. Additionally, I refined algorithmic processes for user verification, supporting the outcome of evaluating computing solutions with proper trade-offs. Remaining course outcomes I aim to demonstrate include professional-quality communication and building collaborative environments, which I will address by documenting my code thoroughly and seeking peer feedback before the final ePortfolio publication.

## Process Reflection

Enhancing this artifact taught me valuable lessons in refactoring and secure design. As I replaced hardcoded credentials with a salted hash mechanism, I faced challenges integrating a cryptographic library and ensuring cross-platform compatibility. Breaking up monolithic functions into smaller classes required careful planning to preserve existing functionality while adding tests. Writing unit tests for the new authentication flow improved my confidence but necessitated additional mocking for input/output streams. Overall, this process deepened my understanding of balancing security, usability, and code maintainability.