**Merrik Wright**  
 **CS499**

### **Software Design & Engineering: Encryption and Decryption System**

**Artifact Description**

**The artifact selected for Milestone One is the Encryption and Decryption System originally developed in CS-405: Secure Coding. This system provides a foundational implementation of AES-based encryption and decryption utilities in C++, designed to protect sensitive information during transmission and storage. For the CS-499 capstone, I enhanced this artifact to improve its modularity, strengthen input handling, and align it more closely with professional software design principles. These enhancements were committed to my GitHub repository in June 2025 and serve as the basis for demonstrating my software engineering growth.**

### **Justification for Inclusion**

**This artifact was chosen for my ePortfolio because it demonstrates practical application of secure software design. Specifically:**

**I refactored the original monolithic functions into modular components, separating encryption, decryption, file I/O, and error handling. This makes the system more readable and easier to maintain.**

**I improved input validation and error handling, ensuring that invalid keys, corrupted ciphertext, or unsupported modes fail gracefully with descriptive messages rather than crashing the application.**

**I added unit-style test cases to validate successful encryption/decryption of sample inputs and edge cases, reinforcing the reliability of the design.**

**I updated documentation and inline comments to clarify the role of each module and provide guidance for future developers or maintainers.**

**These improvements make the codebase more professional, maintainable, and aligned with real-world secure coding standards.**

### **Alignment with Course Outcomes**

**This enhancement aligns with multiple Computer Science program outcomes:**

**Design and evaluate computing solutions using sound engineering principles. By restructuring the code into discrete modules, I applied design principles that promote clarity, maintainability, and testability.**

**Apply secure software development practices. The artifact integrates stronger validation and exception handling, ensuring that unexpected inputs or failures do not compromise security.**

**Communicate technical solutions effectively. Updated comments, documentation, and clear modular function names make the code more understandable for both technical and non-technical audiences.**

### **Reflection on the Enhancement Process**

**Enhancing the Encryption and Decryption System gave me the opportunity to take an early academic project and bring it closer to professional standards. One challenge was deciding how much logic to keep in the main program versus breaking into smaller utilities. Striking the right balance between modularity and simplicity required multiple iterations.**

**Another challenge was implementing structured error handling without overcomplicating the control flow. I achieved this by creating a centralized error handler that reports issues consistently, while still returning explicit failure codes when needed.**

**This milestone reinforced the value of modular design, robust error handling, and clear communication in professional software engineering. It also demonstrated my ability to iterate on an existing codebase and elevate its design quality while maintaining secure practices.**