**Milestone Two Narrative**

For this enhancement, I chose my secure login simulation from CS 405, originally called *ModuleFiveEncryptionApp*. It was a basic console-based C++ project that used a hardcoded XOR cipher to encrypt and decrypt a user file. The original version loaded a file with student info, encrypted the content, decrypted it, and printed the results. It worked fine for the assignment at the time, but it was rough under the hood—everything lived in main(), there was no structure, no error handling, and definitely no real-world encryption going on. I picked this project to enhance because it had a lot of room for growth, and I knew I could use it to really show my progress in clean design and secure practices.

For the enhancement, I refactored the whole thing into a more modular, object-oriented layout. I broke the code into two main classes: FileHandler to deal with file reading/writing, and Encryptor to handle the encryption logic and string parsing. This structure makes the program easier to scale, test, and maintain. I added proper exception handling, debug output to help troubleshoot issues, and cleaned up the file path handling to be more flexible. I also threw in detailed comments and function-level explanations so it’s easier for someone else (or future me) to understand what’s going on.

This enhancement hits multiple Computer Science outcomes. It covers CO2 through clean formatting, meaningful names, and strong documentation. CO4 is shown in the switch from procedural logic to an object-oriented design with thoughtful error handling and structure. CO5 comes through in how I separate logic from data, handle inputs safely, and lay the groundwork for stronger encryption down the line. I’ve already hit CO3 through my other artifacts and plan to address CO1 in my final documentation by improving how I communicate the technical decisions I’ve made across the portfolio.

Working on this reminded me how important code clarity really is. When I first wrote the original, I just wanted it to run. But coming back to it after a few months, it was kind of messy. Refactoring it helped me see how much more maintainable and readable things become when I slow down and plan the design a little more. One tricky part was getting Visual Studio to cooperate with file paths, but once I added a debug print showing the current working directory, it was a quick fix. Overall, this version reflects how far I’ve come and how much more intentional I am about writing clean, secure, and reusable code. I feel way more confident now turning this into a portfolio piece that shows how I think as a developer.