Edwin Jones

CS 405: Secure Coding  
June 7, 2025

Process Summary

I first implemented the encrypt\_decrypt function to fulfill the core requirement of transforming the input using XOR encryption. The original placeholder line output[i] = source[i] was replaced with output[i] = source[i] ^ key[i % key\_length] to ensure each character in the source string was XOR’d with the corresponding character in the key. The use of the modulo operator allowed the key to repeat as necessary when the source string was longer than the key. I added assertions to ensure both the key and source string were non-empty and that the output length matched the input, helping to catch potential issues during development and testing.

Next, I worked on the read\_file function to complete the logic for loading a file into a string. I used std::ifstream along with a std::ostringstream and rdbuf() to read the entire file, including newlines, into memory. To guard against file access issues, I checked that the file was successfully opened before attempting to read it, and I included an error message to inform the user if the file could not be accessed. This ensured that input data was handled robustly and was ready for processing by the encryption function.

Finally, I implemented the save\_data\_file function to correctly output encrypted and decrypted content in the specified format. I wrote the student name to the first line, followed by the current date formatted using std::put\_time, then the encryption key, and finally the transformed data. I verified the output structure by opening and reviewing the generated files.

