**Milestone Three - Enhancement Two: Algorithms and Data Structure**

For this milestone, I enhanced my project2.cpp artifact from CS-410, which was originally a reverse-engineered C++ program simulating a basic client interface with user permissions. The original artifact functioned, but it lacked structure, validation, and modern design principles. All logic was packed into main(), variables had vague names, and there were no reusable components or clear data structures beyond basic control flow. It was a good learning example but not something I’d show in a portfolio as-is.

For the enhancement, now titled EnhancedClientApp, I redesigned the program to demonstrate core algorithms and data structures in a meaningful way. I created a User class to encapsulate client data and used a vector<User> to store and manage dynamic user records. I implemented key algorithms such as input validation, menu looping with a state machine approach, and structured user management logic. Rather than hardcoding behavior, the updated version lets users add and view client records with clear prompts and reusable logic. It also uses constant values and comments to improve readability and maintainability. I kept the program console-based to keep it simple, but the new structure lays the foundation for a GUI or database integration later.

This artifact strongly aligns with Course Outcome 3 (CO3) by demonstrating the design and implementation of efficient data structures (via vector) and control logic. It also supports CO2, showing professional-quality communication with clear comments, naming, and function organization. I added appropriate error handling and meaningful messages to guide users through the interface.

This enhancement helped me practice refactoring monolithic code into modular components and gave me a chance to work more with standard library containers and user-defined classes. It reinforced how helpful clean design and well-chosen data structures are in making code extensible and testable. I now feel more confident using vectors, classes, and structured loops to build real-world tools.

Overall, this enhanced version—now named EnhancedClientApp.cpp—is a huge improvement and a strong example of my skills in managing user input, permissions, and dynamic data in C++. It reflects my growth in both writing maintainable code and thinking critically about the underlying structure of my programs.