The artifact is a C++ program designed to demonstrate SQL injection vulnerabilities and their mitigation strategies. It was created as part of an educational exercise to showcase how insecure queries can be exploited and how they can be improved with parameterized queries, input validation, and proper security measures. The program interacts with an SQLite database, allowing users to query a "USERS" table while highlighting the risks of improper input handling.

This artifact is an excellent addition to my ePortfolio because it demonstrates my ability to identify and address critical security flaws in software applications. SQL injection is one of the most common web security threats, and by enhancing this program, I have shown my ability to recognize vulnerabilities and implement best practices for secure database interactions. This project also highlights my skills in debugging, testing, and refactoring C++ code, ensuring that database queries are executed safely and efficiently.

Several components of this artifact showcase my understanding of algorithms and data structures. The use of callback functions to process query results demonstrates my ability to manage and manipulate data dynamically. The implementation of a secure query function using parameterized queries reflects my knowledge of memory management and efficient string handling in C++. Additionally, the validation mechanisms added to detect and prevent SQL injection attacks illustrate my ability to work with regular expressions and conditional logic to safeguard applications against potential threats.

The artifact was improved by replacing raw SQL string concatenation with parameterized queries using SQLite's prepared statements. I also added robust input validation to filter out malicious patterns before executing queries. Furthermore, logging and error reporting were enhanced to provide better debugging capabilities, and access controls were strengthened to ensure that user authentication data is handled securely.

In terms of meeting the course outcomes outlined in Module One, this enhancement aligns well with my initial goals. I aimed to deepen my understanding of secure coding practices, improve database management techniques, and apply structured programming principles to real-world scenarios. With these improvements, I have successfully met these objectives. As for my outcome-coverage plans, this project has reinforced my commitment to focusing on security-driven software development, and I plan to continue refining my skills in cryptography, authentication mechanisms, and data integrity.

The process of enhancing and modifying the artifact was both challenging and rewarding. One of the key lessons I learned was the importance of thinking like an attacker to identify potential vulnerabilities before they become security risks. I also gained a deeper appreciation for the role of parameterized queries in preventing SQL injection and the necessity of comprehensive input validation. Debugging the database interactions presented challenges, particularly in handling error messages and ensuring that queries executed correctly with bound parameters. However, overcoming these obstacles helped me refine my problem-solving skills and gain confidence in working with databases securely.