# Journal 1-6: Don’t Leave Security to the End

## Critical Reflection

The statement “Don’t leave security to the end” emphasizes the importance of integrating security measures throughout the entire software development lifecycle rather than treating them as an afterthought. Waiting until the final stages of development to address security often results in vulnerabilities that are harder and more expensive to fix. Security should be a core consideration from the planning and design phases through to implementation, testing, and deployment. By embedding security early, developers can create more resilient applications, reduce risk, and build user trust.

## Secure Coding Best Practices

To prevent threats and develop secure code, several steps can be taken:

1. Conduct Threat Modeling: Identify potential threats, attack vectors, and vulnerabilities early in the design phase.

2. Validate Input: Sanitize all user input to prevent injection attacks such as SQL injection or cross-site scripting (XSS).

3. Use Safe APIs: Prefer APIs that avoid common vulnerabilities.

4. Implement Authentication and Authorization Properly: Enforce strong user authentication and ensure that users only have access to the resources necessary for their role.

5. Apply the Principle of Least Privilege: Limit access rights for users and components to only what is strictly required.

6. Perform Regular Code Reviews and Static Analysis: Use tools like Cppcheck or Visual Studio static analysis to detect and address security flaws early.

7. Keep Dependencies Updated: Regularly update libraries and frameworks to patch known vulnerabilities.

8. Use Secure Coding Standards: Adhere to standards such as SEI CERT or OWASP guidelines.

## Planned Example for Project Two

In Project Two, I plan to incorporate unit testing to ensure security is built into the application from the start. By writing unit tests that validate input handling and test for edge cases and misuse, I can detect potential security issues early in the development process. For example, I will implement tests to ensure that the application does not accept SQL control characters in user input fields. This proactive approach will help enforce data validation and safeguard against injection attacks, demonstrating a commitment to secure coding practices from the outset.

By embedding security-focused testing and analysis early, I aim to ensure that the final product is not only functional but also resilient against potential threats.