**6-1 Journal: Don't Leave Security to the End**

Chris Bridges

Southern New Hampshire University

CS 405: Secure Coding

Professor Mill

06 October 2025

The phrase, “Don’t leave security to the end,” infers that security should be incorporated throughout a project rather than introduced at the end as an afterthought. Waiting to add security features at the end of a project could result in a rework of code base sections, additional hours of labor, and financial costs for the company. Planning for security from the beginning creates a project with clearly defined goals and objectives, avoiding the rush to add minimal security right before the project goes to production. Security should be just as important as functionality and user experience.

Several different steps can be taken to prevent threats. Using secure coding architecture choices like least privilege, encrypting sensitive data, and using defense-in-depth techniques can mitigate the risk of attack from the top level. User stories can be used to identify potential areas of attack or vulnerabilities. Using secure coding techniques, such as input validation during code creation, can reduce the risk of SQL injection and buffer overflow. Use dependency checks to ensure libraries are kept up to date and vulnerabilities are patched quickly. Static and dynamic unit testing should be part of the development process, too. Static testing finds flaws and vulnerabilities without executing the code, while dynamic testing runs the code with different inputs to ensure it behaves correctly and safely. Together, both forms of unit testing will help catch issues before the code reaches production.

One example I intend to add to my project two presentation is input validation. Input validation is something done at the software engineer level and can be added as the code is created. Ensuring that input validation is used while creating the code will keep the engineers in a security-focused mindset while developing, and ensure security isn’t left until the end. By validating user input early in the development cycle, harmful data can be kept from databases and application logic. This can stop common attacks like SQL injections and reduce debugging time. Input validation works hand in hand with automated testing since both static and dynamic testing can be used to confirm input validation rules are enforced.