**Module 6 Journal: Don’t Leave Security to the End**

**Justin Hancock**

**CS-405 Secure Coding**

**10.08.2024**

“Don’t leave security to the end” means that security should be an integral part of the software development process from the very beginning rather than being an afterthought. Waiting until the final stages to address security can lead to vulnerabilities, increased development costs, and delayed project timelines. Incorporating security into each stage of development helps identify and address potential issues early, resulting in a more secure and reliable product.

To prevent security threats, we might consider the following steps throughout the development process:

* Identify potential security threats and vulnerabilities early in the design phase to understand how an attacker might target the system.
* Implement input validation techniques to prevent common attacks like SQL injection and cross-site scripting. Validate all user inputs against expected data types, lengths, and formats.
* Conduct regular peer reviews to identify and fix security weaknesses in the codebase.
* Use static analysis tools to automatically detect common security flaws in the code before deployment.
* Use strong authentication mechanisms, like multi-factor authentication, and implement role-based access control to restrict user access to only necessary resources.
* Regularly review and update external libraries and dependencies to minimize the risk of vulnerabilities within third-party code.
* Protect sensitive data using strong encryption algorithms for both data at rest and in transit.

An example of integrating security best practices into the presentation could involve automated static code analysis. This approach would automatically scan the code during development to detect potential vulnerabilities like buffer overflows, insecure API usage, or hard-coded credentials. By incorporating static code analysis early in the development cycle, developers can identify and address security issues proactively, rather than discovering them after deployment. Using this method in the policy demonstrates how automation can support ongoing compliance and security throughout the development process, aligning with a defense-in-depth strategy.