James Galbreath

CS-405

6/18/25

All too often software engineers take on the tedious task of developing well-functioning code to specific parameters but overlook the fact that if it is not secure it is ultimately not well functioning. Regardless of what it can perform, a lack of security renders its functionality null and void if the application calls for any semblance of security. Software designers must keep security in the forefront of their minds and build a program from the ground up based off secure coding practices.

Steps to follow these standards could be included like validating all input, and applying the principle of least privilege. Some security tools like static analysis and automated tests should be integrated into the CI/CD pipeline to catch vulnerabilities early. Encrypting data at all stages and conducting regular code reviews further ensures a proactive, built-in approach to secure software development.

One example I will include in my Project Two presentation is the use of **automated unit testing focused on input validation**. By writing tests that check how the application handles invalid, unexpected, or malicious input, I can catch potential vulnerabilities like buffer overflows or null pointer dereferencing early in development. This ensures security is built into the code from the start, rather than waiting for issues to surface later.