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

**Ryan DeBraal**

**CS-405-X6389 Secure Coding 21EW6**

To a capable developer I would *hope* this writing prompt would be self-apparent. It is much easier to build code with security as a guiding principle from the beginning rather than try to bootstrap a secure coding philosophy after a security incident has occurred.

The best way to ensure you are building a secure application is to assume that every line of code you write is potentially introducing a new security flaw which could be exploited. With this “paranoid” mindset, a developer becomes much more meticulous with how they use public/private accessors, data types and validation of data from external sources (such as user input). As Benjamin Franklin said: “An ounce of prevention is worth a pound of cure.”

When working on Project Two, I plan to utilize the same *rules* I’ve used throughout my software development career.

These rules include:

* Sanitize all user input, whether that be through RegEx or some other means. Verify that the data coming from an external, and therefore uncontrollable source, is valid and doesn’t contain any malicious injection code or cross-site scripting (XSS)
* Use a “fail-fast” philosophy, meaning fully validate data for any possible state it may be in before you even attempt to use it. Do this on repetitive layers to satisfy Defense in Depth (DiD) requirements.
* Think like the “bad guy”, look at your code from the perspective of someone who wants to break it. Whole would you go about subverting your code? Refactor accordingly.
* Test, test, and retest!