This course has helped me to clearly see the need to adopt a secure coding standard. It often is not enough to just say “let’s make the code secure”, but instead it helps to have a documented standard by which you can measure code to ensure that it is secure. Having such a standard to guide the coding process also ensures that security is considered even before the software begins being developed, and that security is considered throughout the development lifecycle. Adopting this posture takes a more aggressive stance against attackers instead of being reactionary in nature. Doing so creates a secure software from beginning to end and saves time, money and resources for any company.

Taking an active stance in security instead of reactionary also leads to the process of evaluating the most like avenues of attack for the software being developed. This evaluation process then helps the company to understand the potential risk that is exposed at each point in the software and the potential cost of failing to secure the code. By developing according to the coding standards and anticipating attack avenues, it should quickly become clear that preventing attacks will also mitigate any costs that a successful attack may have incurred.

This leads to the concept of a Zero Trust policy in which no user, external or internal, and no point of entry should be trusted. Authenticating users in multiple ways (MFA) and ensuring that users are able to view only content that are authorized to view, and then taking steps to record and account for every action taken on in a system provides the framework for the Zero Trust policy. This Triple A defense promotes a healthy and secure environment not only for the company who owns the software, but for the users who utilize the software, providing security for both parties.

To implement such security policies, the company must first choose a coding standard, and ensure that all developers understand the need to follow that coding standard. Then code should be developed with peer review and unit tests performed throughout the development lifecycle and quality assurance testing performed only at the end when the developers are comfortable with their security. This helps to move the testing process along more quickly and prevents the team from having to go back and understand security holes that may exist in the system.