**Secure Coding and Security Policy Reflection**

One of the most important lessons I’ve learned during this course is the value of adopting secure coding standards early in the development process. Too often, developers wait until the end of a project to think about security, which can lead to major vulnerabilities being missed. Secure coding standards help developers write safer code from the start. For example, using input validation, safe memory management, and proper error handling can stop issues like buffer overflows or injection attacks. If security is part of every coding decision from the beginning, it becomes a natural part of development instead of an afterthought.

Another key topic is evaluating risk and weighing the cost-benefit of mitigation. Not every risk is equal, so it’s important to assess which ones have the highest potential for damage and prioritize fixing them. For instance, a vulnerability that could expose customer data would be more urgent than one that causes a small performance issue. Using tools like risk matrices can help us rank threats based on how likely they are and how severe the impact would be. This helps teams focus their time and resources on the most important issues and avoid wasting effort on low-risk concerns.

Zero Trust is another concept that stood out to me. At first, it may seem extreme to assume that no one inside or outside the network can be trusted by default. But in today’s world, with insider threats and advanced cyberattacks, this approach makes sense. Zero Trust means always verifying users, devices, and applications before granting access. This includes using multi-factor authentication, strict access controls, and constant monitoring. It can be more work for users, but it greatly increases the overall safety of the system.

Lastly, implementing and maintaining strong security policies is critical for any organization. Security policies give clear instructions on how to handle sensitive data, access systems, and respond to threats. These policies must be realistic, easy to follow, and updated regularly to keep up with new technologies and threats. Recommendations for good policies include regular training for developers, regular audits of system security, and the use of automated tools to detect vulnerabilities.

In conclusion, security should be a part of every step in the development process not just the final phase. By adopting secure coding standards, evaluating risk properly, following the Zero Trust model, and using clear security policies, we can build systems that are both functional and safe. These practices protect not just the code, but also the people and businesses who rely on it.