Having a secure coding standard is extremely important and essential in developing a strong software. One important lesson I have learned throughout this course is that security should be at the front of developers minds and not left behind. The coding standers for example show that prevention over patching is a better starting point for developers to follow. This will make sure that input validation is put in from the start and not left behind to rust. This approach to design makes sure that vulnerabilities as whole will be reduced, and you will also save time and cost for yourself down the line. Looking back in my coding life I know see how many bugs and issue I could prevent from the start

A crucial part of secure software development is that evaluation and assessment of risk. Every choic you make has its pros and cons in this assessment. Helps you choose the more important ones to look at. As discussed in our course readings, even simple mitigations like input sanitization, least privilege enforcement, or automated testing with tools like static analyzers can drastically reduce attack surfaces with minimal cost. For instance, if the risk of SQL injection is high, using parameterized queries early in development can be a low-cost fix with a high payoff in terms of protection.

Zero trust was another important topic that I personally had alot of fun looking at. Understanding the idea that “no none is safe” was interesting to read about. This thought now demands that during Develpment I can see why certain choices are made. It makes developers abandon the idea that the system they have is safe. If something is easy to break on the outside, you have to make sure the inside is even stronger. This concept as a whole will make me look deeper into how things work. My favorite example I found is thinking if APIs are safe you have to have rate limiting services which need to be kept to a high standard.

Lastly security policies are good for guiding teams in authentication, data storage, encrpytion and so much more. Learning about the DevSecOps life cycle will let me make smarter and more inclined choices going forward. These coding practice exist for a reason and there should be no reason people do not use them.