Greg Monti

April 23, 2025

CS-405-Secure Coding

**Journal: Portfolio Reflection**

Looking back on everything I’ve learned in this class, I have to say, I definitely came into it thinking secure coding was something you tacked on at the end. Like, “Let me just get it working first, and I’ll worry about security later.” But now? Totally flipped perspective.

One of the biggest things that stuck with me was how important it is to adopt a secure coding standard from the start. Not as an afterthought, not as some final checkbox after deployment, but baked in right from the first line of code. I used to think that kind of rigor would slow me down, but what really happens is it saves you from bigger headaches later. In one of our early readings, there was a section about how vulnerabilities introduced during early development stages become exponentially more expensive to fix the further you go. And it’s not just about cost, it’s about keeping trust, protecting users, and just doing the job right.

That ties directly into evaluating risk and cost-benefit of mitigation. I used to think security fixes were “nice-to-haves” unless something was clearly broken. Now I think in terms of threats and impact. We worked on that threat matrix for Green Pace, and it helped me see that not all risks are created equal. Some things, like SQL injection or buffer overflows, are so critical that they need immediate attention. Others might not justify the cost to mitigate fully, but at least you make an informed call. The reading that talked about STRIDE and DREAD helped me figure out how to actually weigh risk in a structured way, not just guess.

Now let’s talk about Zero Trust. At first, the concept annoyed me a little, like, wow, sounds kind of paranoid. But the more I thought about it, the more it made sense. We live in a world where attackers don’t just waltz in through the front door. They exploit insiders, compromised credentials, misconfigured firewalls, whatever crack they can find. Zero Trust is basically saying, “Prove yourself at every step, every time,” and yeah, that is a little paranoid, but it's also smart. It’s like saying, “I trust you, but I’m still checking because I have responsibilities.” And as developers, we do have responsibilities.

Finally, implementing and recommending security policies felt super intimidating at first, but once I broke it into categories, like encryption, Triple-A, access controls, and automation, it became manageable. I actually liked how our final presentation for Green Pace pulled it all together. It felt like I was building a security culture, not just plugging holes. Having policies in place helps the team stay consistent. No one’s guessing what the standard is, because it’s written out and part of the process.

All in all, I’ve learned that secure coding isn’t just for “security experts”. It’s part of being a responsible developer. And now that I’ve got this mindset, I can’t unsee the risks in code I used to think was fine. It’s a little stressful, sure, but it also feels like growth.