Trying to figure out why someone might cause a security problem should be something we do often during development, even if we don’t always know the exact reason. People who hack systems can have all kinds of reasons—some do it for money, some to make a statement, and others just because they enjoy breaking things. Even if we can’t always tell what their motive is, thinking about it helps us take a closer look at the weak spots in our system and understand how someone might try to take advantage of them.

I’ll use this by not only looking at the technical problems, but also by asking myself things like, “What would someone try to get out of attacking this?” or “If I wanted to break into this system, what would I target?” Doing this helps me move away from just going through a list of security tasks and instead focus on how someone might actually try to misuse the system. It also helps me think about the worst things that could happen early in the process, instead of waiting until after the app is already done.

I’d tell them that even if we don’t always know exactly why someone would want to attack our system, we should still act as if it could be a target. I would explain that they should always ask themselves how someone might try to interfere with the system or steal information, and what the attacker would gain from doing so. Thinking in this way helps them start to see the system from an attacker’s perspective, which makes it easier to recognize weaknesses and spot potential problems before they actually happen.

One example I’ll use in my Module Eight reflection is the idea of zero trust. This means you don’t automatically trust anything, whether it’s inside or outside the system. You treat everything like it could be a threat until you know it’s safe. This connects to the idea of looking for motive, because zero trust is all about expecting that someone out there will try to take advantage of your system—whether it’s for money, access, or just to cause trouble. It’s a way of thinking that matches with always keeping motive in mind, and it fits with the bigger goal of using secure coding practices early and checking for risks on a regular basis instead of waiting until something goes wrong.