* Adoption of a secure coding standard, and not leaving security to the end
  + This is important. My main reasoning is why do something twice when you can do it right the first time. Now everyone makes mistakes, but performing secure practices from the get go easier than going back and trying to add them. Adding could cause major code revision and take up more time. No point. So outlining even a basic secure coding standard that can be built on as the project unfolds is better than nothing at all.
* Evaluation and assessment of risk and cost benefit of mitigation
  + This is important, but it’s not something I’m good at and would require more help from others to better understand and perform.
* Zero trust
  + Zero trust falls in line with Least Privilege. This is something we should all practice. It may be easier to give open range to something, but we all know what happens when we do that. People try to break stuff, either for fun, by mistake, or with bad intent.
* Implementation and recommendations of security policies
  + I think towards coding standards/policies we need to look at past projects and see where we’ve had our most common bugs. This is where I’d start for direct coding. Other things we need to start from a more holistic point of view. How are things as it currently stands, what network security is in place. Do we know who owns what. What areas are we lacking in. How can we change those? This will tell us the depth at which changes need to be made and we can take into account repeated missed areas for higher priority policies.