**Module Eight Journal**

Preventing threats and prioritizing security from the beginning of the development lifecycle are important for making sure that data is kept safe. It is helpful to begin by cultivating a culture of secure coding in the workplace so that it is automatically in the developer’s minds, and a secure coding standard should be adopted that defines a set of rules and guidelines to follow during development. This will ensure that there are clear systems in place to prevent, detect, and eliminate issues. It is crucial that security is considered from the very beginning rather than waiting until the end. Waiting might be tempting, as implementing security early can incur cost, take up time, and requires the correct knowledge. However, waiting until the end makes it more difficult to efficiently include security as the code is already structured. This will end up costing more in both time and money because the code will need to be rewritten and refactored in different areas. Not waiting until the end protects reputations, trade secrets, sensitive data, etc. and helps to make sure that a high quality product is produced and serious consequences are avoided.

Some risks of mitigation are protecting the wrong areas of a system, incorrectly putting security into place, or not effectively eliminating a threat once found. This could be dangerous as it gives a false sense of security. Other risks may be missing deadlines or paying more upfront costs. On the other hand there are many more benefits of mitigation and far outweigh the risks. It makes a company seem more reputable because the public will see that it is taking the time and using the resources to protect sensitive data and keep in compliance with best practices. It is an investment into the future, as putting security in place now can help avoid potentially disastrous breaches down the road with serious consequences.

Zero trust is a security approach that emphasizes practices like verification on every device, user, or app and treating all resources as external and therefore untrusted. This is in contrast to more traditional methods that focused on keeping a secure perimeter but using little to no protection once access is granted, similar to protecting a castle with a moat. There are at least five pillars of trust to focus on. First is device trust, as a device needs to be known before it can be trusted. Second is user trust, like making sure that passwords are not the only form of user verification. Third is session trust, which focuses on the principle of least privilege. Fourth is application trust to make sure that apps needed for the digital workspace are secure, and fifth is data trust to make sure that users are using correct data. Building trust across these pillars help make decisions on whether or not to grant access.

Security policies should be implemented all throughout the development lifecycle or the DevOps pipeline. Automation tools can be used for tasks like static analysis, penetration testing, endpoint detection, and maintenance. Following policies like encryption and Triple A is also advantageous. Defining these points in a security policy and enforcing its use will help create a secure and trusted system.