# CS 405 Project Two Script Template

Joshua Hampton

CS405

**https://youtu.be/aCs-IAeXzj0**

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | This presentation is to be used for training purposes with regards to the Security Policy by Joshua Hampton |
| **2** | Defense in Depth is the practice of a layered security defense pattern. This begins with the developer by ensuring secure coding practices throughout the development process. |
| **3** | The highest priority threat is that of SQL injection, not only is this the most common attack vector, but it can also be the most damaging as this could compromise data security and integrity. Phishing is a very likely threat as this can come in many forms and once a user’s credentials are compromised it may be difficult to tell it has happened. A low priority threat would be from an outside source attempting to brute force their way into the network through password guessing. This can easily be mitigated through account lockouts and MFA. Unlikely threats include an internal criminal actor such as an employee who is attempting to compromise the network security from within. |
| **4** | The ten security principles are Validate Input data, this can prevent the aforementioned SQL injection attacks. Next is Heed compiler warnings, this prevents errors within code that can cause small bugs or large security vulnerabilities. Next, Architect and Design for security policies. This is to ensure that security is implemented throughout the entire software development lifecycle. Next is the policy of keeping it simple, this is to ensure the code is easily readable and maintainable. Default deny ensures only users which need access to the program have it and no one can access any portion of the app who is not intended. Adhere to the Principle of Least privilege is an extension of Default deny in which no user shall have access to any data that is not required for them to do their job. Sanitize Data Sent to other systems ensures user data is not compromised. The use of effective Quality Assurance Techniques ensures developers keep security and application integrity in mind at all stages of development to ensure bug free applications. Adoption of a security policy ensures all developers are following the same standard and are held to that standard. |
| **5** | Each standard represents a secure coding practice to adhere to and ensure secure and bug free code which cannot be exploited by malicious actors. |
| **6** | Encryption is an important part of secure coding. Sensitive data should be encrypted at all stages of use. Encryption at rest refers to data that is being stored on permanent devices such as hard drives. Encryption in flight refers to data that is being transmitted either over the network, over the internet, or in a drive that is being transported to another facility. Encryption in use refers to the data that is being currently used and stored on the machine in temporary memory such as RAM. |
| **7** | Triple-A policies refer to Authentication, Authorization, and Accounting. The ensures all users are authenticated and allowed to access the network information. Authorization ensures the authenticated user is supposed to have access to the information they are attempting to access. Accounting ensures a log of user access so that in the case of a potential security breach it is know what was compromised and by what account. |
| **8** | CPPCheck can be used in addition to IDE compiler warnings to check for security flaws and potential coding errors. |
| **9** | Automation in the DevSecOps lifecycle ensures that all tests are run and developers adhere to the security policy. |
| **10** | Dynamic Application Security testing can be integrated into the software development lifecycle to automatically test for new and emerging security risks in software. Other static testing tools such as CPPCheck can be used to check for coding syntax errors and insecure coding. |
| **11** | All security risks should be taken seriously, but technical debt and current projects must continue to be developed. Using the risk assessments provided in the security policy can help to prioritize certain flaws over others to ensure high risk security flaws are addressed immediately. |
| **12** | There are security principles which were not adequately addressed and security policies should be added in future iterations to address this. |
| **13** | Future iterations should add secure coding standards to demonstrate Encryption, Zero trust design, and API integrtions. |