# CS 405 Project Two Script Template

Complete this template by replacing the bracketed text with the relevant information.

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | Security Policy Presentation. |
| **2** | Overview, defense in depth.  This security policy outlines several standards and policies that could help define requirements for applications to potentially improve security. This policy could help with implementing several layers of security for a defense in depth strategy. |
| **3** | Threats matrix.  Priority threats could be identified based on the potential impact on the system, the likelihood of occurrence and the remediation cost. The higher the severity, likelihood, and remediation cost, the higher priority the threat could be. Lower priority threats should potentially also be resolved, but might not pose as much of an immediate threat to the system. |
| **4** | Ten principles.  Some examples of how the principles and standards align could be the following.  Validate Input Data could relate to String Correctness, SQL Injection, and Input Validation  Keep It Simple could relate to Data Value, and Resource Management  Default Deny could relate to Memory Protection  Practice Defense in Depth could relate to Exceptions, and Concurrent Safety  Use Effective Quality Assurance Techniques could relate to Data Type, and Assertions  Adopt a Secure Coding Standard could relate to Date Type, and Input Validation |
| **5** | Coding standards.  The standards were ranked by priority, the highest priority standard being at the top of the list. Priority was determined based on severity, likelihood, and remediation cost.  Standard five, high severity, likely likelihood, medium remediation cost, high priority.  Standard ten, high severity, likely likelihood, medium remediation cost, high priority.  Standard one, high severity, unlikely likelihood, medium remediation cost, high priority.  Standard three, high severity, possible likelihood, high remediation cost, high priority.  Standard four, high severity, possible likelihood, high remediation cost, high priority.  Standard seven, high severity, possible likelihood, high remediation cost, high priority.  Standard nine, high severity, possible likelihood, high remediation cost, high priority.  Standard two, medium severity, likely likelihood, medium remediation cost, medium priority.  Standard eight, medium severity, possible likelihood, medium remediation cost, medium priority.  Standard six, medium severity, unlikely likelihood, low remediation cost, medium priority. |
| **6** | Encryption Policies.  Encryption at rest could involve encrypting stored data. This policy could apply to all data stored on organizational servers, cloud storage, and backup systems. Encryption at rest can be important for preventing unauthorized access to data.  Encryption in flight could involve encrypting data that is being transferred over networks. This policy could be applied to all data transfers within and outside the organization’s network, including data shared utilizing APIs, web services, and email communications. This policy could be important for protecting data during transit.  Encryption in use could involve encrypting data that might be actively being utilized by applications. This policy could cover all data in the process of being utilized by an application within the organization. Encryption in use should help with protecting data from unauthorized access and or leaks during operations. |
| **7** | Tripple-A Policies.  Authentication could involve verifying the identity of a user, process, and or device, often as a prerequisite to granting access to system resources. This policy could apply to all system logins and application accesses. Authentication could help with trying to only granting access to systems and data to authorized individuals, which can help with preventing unauthorized access.  Authorization could involve trying to only grant access to authenticated users for specific resources and actions. This policy could apply to user roles and permissions for the system. Authorization could help grant access permissions based on job functions, which could potentially help limit the risk of unauthorized system access.  Accounting could involve tracking and logging user activities for the purpose of analyzing resource utilization and potentially detecting anomalies. This policy could apply to all user activities, database changes, and potentially sensitive data access. Accounting could help with monitoring user actions, policy compliance, and potentially detecting security incidents. |
| **8** | Unit Testing.  Collection Smart Pointer Is Not Null  This test should check that the null pointer managing the vector is properly initialized and not null after the setup. |
| **9** | Unit Testing.  Clear Erases Collection  This test should check if the vector is correctly cleared, which could be important for preventing memory leaks due to remaining data. |
| **10** | Unit Testing.  At Throws Out Of Bounds Exception  This test should check if accessing beyond the size of the vector correctly throws an exception, so that memory outside the allocated range is not accessed. |
| **11** | Unit Testing.  Reserve Lower Than Size Reduces Capacity  This test should check if attempting to set the vector capacity to a value less than its current size does not decrease the capacity. |
| **12** | Automation Summary. |
| **13** | Tools.  The DevSecOps pipeline is a security focused strategy integrated within the software development lifecycle, considering security measures from the design stage to the production stage.  Some external tools that could be utilized for DevSecOps are SAST for secure coding, DAST and SCA for testing, SIEM for monitoring and incident response platforms for automating security checks and responses. |
| **14** | Risks And Benefits.  Potential problems: security risks, unexpected application behavior, and potentially unsecure data.  Potential solutions: Integrate defense in depth to secure the application, allocate resources to protect against potential threats, prioritizing ones that could be substantial security risks, are likely to occur, and costly to mitigate, and follow the security policies outlined on the document.  Potential risks of waiting: security incidents could occur, and if the application is not very secure, these incidents could cause loss of data confidentiality, and or application functionality.  Potential benefits of acting now: protecting application availability, and data integrity and confidentiality. |
| **15** | Recommendations.  The policy document could be updated regularly, to keep up with the latest potential security threats.  Staff members could regularly take security training to potentially better implement the standards and policies outlined in the document. |
| **16** | Conclusions.  All standards outlined in the security policy document could potentially be followed for application and data security. These standards could be prioritized based on the potential severity of the security risk, the likelihood of occurrence, and the cost of mitigation. |
| **17** | References. |